

SPECTRUM

VILÁG 21.

+4 oldal ENTERPRISE

49 Ft



A következő Spectrum 48K (S151-S158, S160) és Spectrum 128K (S159) kollekciók is megrendelhetők utánvétellel a Spectrum Világ címén (Budapest, Pf.: 363, 1519) keresztül. Egy kollekció ára ÁFA-val és postaköltséggel együtt 300,- Ft. Megrendeléseikben kérjük a kollekciók sorszámát pontosan megnevezni!

S151 A: Abracadabra I-II / The Duct / Habilit / Dea Tenebrarum / Swat!
B: Robotscape / Circus Games (part 1-6)

S152 A: Vampire's Empire (+ part 3) / Technocop (+ part 3) / Echelon (part 1-3) / Fire & Forget
B: Hellfire Attack (part 1-7) / Shoot Land (+ part 4) / Shoot Out / Mad Mix, the Pepsi Challenge

S153 A: Captain Blood / Task Force / Robocop (+ part 4) / Colditz / Go Into Second Gear / Manhattan
B: Aquarius / Carrots from Space / Sheep Dog / Norman Castle / Jewels of Darkness I-II-III / Estimator Racer / Soft & Cuddly

S154 A: Chicago 30's / Arctic Fox / Championship Sprint (+ editor) / Cavern Fighter
B: Bear A Grudge / Corn Cropper / Fire On the Water / Dreadnoughts / Gerry the Germ / Streethawk

S155 A: Turbo Boat Simulator / Exploding Fist + / Rally Simulator / Motor Massacre (+ part 3) / Tuareg
B: Death Stalker / International Rugby Simulator / Jocky Wilson's Darts Challenge / Gi Hero / Peter Pack Rat / Traz

S156 A: Vindicator (part 1-3) / Return of the Jedi (Domark) / Pacmania / Bounty Bob Strikes Back
B: Four Soccer Simulators (11 A Side, Indoor Soccer, Street Soccer, Soccer Skills) / Strippoker 2

S157 A: Roy of the Rovers / Artura / Sabrina / Chubby Gristle / Secret Mission
B: Rock'n Roller / Wells Fargo / Lightning Simulator / Free Climbing (part 1) / Professional Skateboard Simulator / River Raid / Repulsar

S158 A: Soapland (level 2 - kéri a level 1-ből kimentett állást) / Foxo Fights Back / Blade the Warrior / Babaliba / Gyron Atrium / The Rocky Horror Show
B: Skooidaze / Rocky / Strontium Dog the Killing / Engineer Humpty / Wormy Wawe / Chiller / Yacht Race

S159 (128K) - A: The Raven 128 / Ghostbusters 128 / The Pawn 128

B: Stalker 128 / Tank 128 / Bionic Commando 128

S160 A: Sam Stoat / Jet Set Willy II. / The Saga of Erik the Viking / Witch's Cauldron / Gremlins the Adventure / Airwolf
B: Mr. Mouse / Mc. Wash / Sheer Panic / 3D Star Wars / Race Fun / Speed Duel / Joust! (LOAD CODE) / Androids / Kong II (Strikes Back) / Fighting Warrior / QS-Chess

KAISER PÉTER, budapesti Olvasónk küldte be ezt a szép borítótérvet. Sajnos ilyen – előre meg nem rendelt – munkáért pénzbeli honoráriumot küldeni nem tudunk, ám a munka minősége alapján feltétlenül úgy gondoltuk, hogy közöljük a SpV.-ban. Péter, csak így tovább!



MINDEN KEDVES OLVASÓNKNAK KELLEMES KARÁCSONYI ÜNNEPEKET, ÉS EREDMÉNYEKBEN GAZDAG BOLDOG ÚJ ESZTENDŐT KIVÁNUNK!

Újra itt vagyunk! Nem akarjuk dorgálni a Tiszteit Olvasót, elég sok dorgálást kaptunk, főként a 20. szám bevezetőjének utolsó mondatáért, de hát ez az igazság.

Néhány dolgról dióhéjban:

• **Játéklismertető:** Több, néhány mondatos ismeretelőre is igény mutatkozott, ezt most megpróbáltuk összefonni sok-sok apró cheat-tel, mindent bale alapon, elvégre ez a szám kerül a karácsonyfa alá.

• **Ismeretlen nyelvek:** Sok-sok dicséretet kaptunk a két programnyelv (*Micro-PROLOG, 'C'*) folyamatos ismertetéséért, hiszen ezekről még szinte sehol sem jelent meg leírás, ezért folytatjuk is.

• **128K:** A legfontosabb – 128K-s gépre vonatkozó – információkon már túl vagyunk, 3 csatornás hang – BASIC listákkal a továbbiakban nem töltjük meg az oldalakat, ennek sok értelme nincs. Amennyiben összegyűlik 1 oldalnyi, kimondottan 128K orientált információ, úgy azt közöljük.

• **Gépi kód tanfolyam:** Ilyen címszó alatt megszű-

nik, szerepét a programozástechnika veszi át a továbbiakban.

• **Rejtvény:** A többség nem kéri, ezért az itt található levelezési rovat a következő számtól oda telepszik. Ez utóbbit viszont annál többen hiányolták eddig.

• **Térképlap:** Sajnos nem gazdaságos a pótlap legyártatása, most a Karácsonyra való tekintettel ezt az A/4 lapot még közreadjuk, ám a továbbiakban a térképeket ismét beszorítjuk a belső oldalakra. Ezzel is azt szeretnénk elérni, hogy a január 1-n esedékes jelentős nyomdai áremelkedéssel egyidőben mi ne emeljük a kladvány fogyasztói árát!

Energilánk továbbra is véges, a jelenlegi feltételrendszerben sokkal többet nyújtani nem tudunk. Ennek ellenére nem érezzük, hogy az előző okfejtésünk óta lényeges változás állott volna be az eladott példányszám vonatkozásában. Nagyon elképzelhető, hogy 1990-ben ismét csökkentenünk kell a példányszámot (a jelenlegi 12.000-ról), ill. a megjelenési gyakoriságot egyaránt.

Last Ninja 1 hol vagy?

Tiszteit SpV! Szeretném megkérdezni, hogy a THE LAST NINJA (1) c. programot végülis átirták a Spectrumra, mert sok újságban ill. katalógusban szerepel. Példának okáért:

- a Spectrum Világ 4. számának 2. oldalán

- a Sinclair Spectrum Játék és Program c. könyv 4. részének mellékletében.

- stb.

Úgyhogy ezért kérdezem, mert nem tudom, hogy végülis átirták vagy nem. Nekem a II. része nagyon tetszett, és C64-en láttam az első részét is, és nagyon szeretném megszerezni. Ezen kívül kérném megkérdezni a SYSTEM-3 nevű software forgalmazó cég címet, mert egyik újságban sem találtam. Farsadozásukat előre is köszönöm. Várom a választ.

Iff. Gyóni Péter – Bp.

Át is írták, meg nem is, szóval mi sem tudunk többet, annyit keveredés volt a program Spectrumos változata körül. A Spectrum Világ 4. száma ill. az említett könyv 4. része kb. azonos időben jelent meg, ekkor még az angol sajtó nagy ovációval hirdette, hogy a program hamarosan megjelenik, a mi hittünk nekik. Ez azonban sajnos elmaradt, pontosabban megdöbbenve tapasztaltuk, hogy előbb utóbb végül is a 2. rész látott napvilágot, annak ellenére, hogy az angol sajtó sok-sok screen-t is bemutatott az 1. rész Spectrum verziójáról. Nos, madarak azt csirpelték, hogy a programot hazai programozók fejlesztették, a valami új programozási módszert akartak kidolgozni ebben a programban. Ezek szerint a fába beletört a fejsze...

A SYSTEM 3 címet mi sem találtuk, helyette itt a telefon-számuk: Anglia - 01-866-5692

SpV

Egy a sok közül

Tiszteit SpV! Örültem, hogy megküldték nekem az SpV 19. számát. No de nem ezért írok most önöknek.

Az Airborne Ranger-ről van szó. Miután ledobtam a három utánpótlásosomagot, megjelenik egy nyíl és game over. Elolvastam a 19. rész ismertetését, de sajnos nem jutottam vele semmire. Jó lenne, ha néhány sorban megírnák nekem, hogy mit kell csinálni, mert ideges vagyok amiatt, hogy felvettem egy olyan programot, ami aig fért fel egy oldalra, és nem tudok vele mit kezdeni. Más. Lehet, hogy tudok küldeni egy pöket a Cybernoid II-hoz.

Gombos Bertalan, Bonyhád

Az Airborne Ranger annál bonyolultabb játék, hogy néhány szóban bármilyen ötletet is tudnánk hozzá ajánlani. Ez a levél egy a sok közül, amelyben Olvasóink hasonló problémákkal keresnek fel bennünket a mai napig. A 17. szám borítóján már egyszer tisztáztuk, hogy inkább a Spectrum Világ-ot írjuk, minteam, hogy napi 10-12 órában ilyen témájú levelekre válaszoljunk, mert akkor soha sem jelenne meg a Spectrum Világ. Az Airborne Ranger egyszer úgy is megárdemni, hogy leiratát közöljünk róla, főleg azért, mert valóban elfoglalja egy 60 perces kazetta egy oldalát. A Cybernoid II-hoz nekünk is van POKE-unk: POKE 36687,0 (sérthetatlenség), de az említett 20. számban CHEAT-et is közöltünk hozzá!

SpV.

SAM-betűnő kérés

Tiszteit SpV szerkesztő! Valamelyik SpV. számban olvastam a legújabb ZX-Spectrum gépről, a ZX Spectrum SAM-ről. Érdeklődni szeret-

nék, hol, és hogyan lehet beszerezni ezt a gépet. Előre is köszönöm.

Németh Ádám, Gyál

Tisztáztuk, a SAM nem ZX Spectrum altípus, egy teljesen egyedi számítógép, amely a ZX-Spectrum ROM-programjának külön betöltésével hajlandó ZX-Spectrum-ként is működni. A gépet tudomásunk szerint továbbra sem lehet üzletben megvásárolni, csak levél útján van lehetőség megrendelni, közvetlenül, a MILES GORDON TECHNOLOGY címén keresztül. Ezt a címet már közöltük a SpV 18. számának borítóján.

SpV.

Elégedetlen gyerekek

Szerkesztőség! Egy kérésünk lenne magukhoz: Szeretnénk, hogy közöljék a legközelebbi SpV-ban a Last Ninja 2-örökletét! Köszönjük!

És ez az irányítás? semmi joystick, vagy talán valami kézreésső gombok.

Egy ilyen programnál Spectrumos gyerekek!

Diós Sándor, Budapest

A Last Ninja 2-nek nem öröklete, hanem örökletei vannak, vagyis mind a 8 szintnek más címen kell beállítani az örökletét: Level 1: 36578,0 - Level 2: 35993,0 - Level 3: 36571,0 - Level 4: 36514,0 - Level 5: 36393,0 - Level 6: 36822,0.

Az irányítást pedig tessék elfogadni, ez van!

SpV.

Hol van az a hang...

Csöszí Tibor, Veszprém

A 128K gépek tulajdonosainak általános problémája, hogy RF ószakoztatás nélkül a hang és a kép nem esik egybe a televízió készüléken. Ez főként azoknál a gépeknél

fordul elő, amelyet valaki közvetlenül Angliából hozott meg, vagy Ausztriából, ill. NSZK-ban olyan kereskedőtől vásárolta, aki közvetlenül a szigetországból rendelte az árut. A probléma oka az, hogy amíg nálunk az 5,5 MHz-es differencia elfogadott a kép és hang között, addig ez a szigetországban 6,5 MHz, innen adódik az eltérés. Természetesen az UHF modulátor megfelelő beavatkozással átkalkítható, de az is megoldás, ha Londonban vásárolunk a géphez televíziókészüléket. Ez utóbbi esetben rádióként használhatjuk itthon a TV-t, svagy némafilmeket nézhetünk...

SpV

Embléma javaslat

Puiz András, Budapest

Annyi idő után most már mi is szeretnénk volna megtalálni igaz 'hangunkat', ezt célozta a COMMODORE VILÁG-gal egyidejűleg beindított új fejléc is, amelyet most már szeretnénk véglegesnek tekinteni. Akik nem iamerték volna fel, az 'S' betű a SINCLAIR-leiratból lett átvéve. A beküldött – saját tervezésű – 'S' betű szép, ezért a következő számban megmutatjuk az Olvasóknak is. Ettől függetlenül – első-sorban a nyomdai munkák egyszerűsítése érdekében – a jelenlegi változat mellett maradunk.

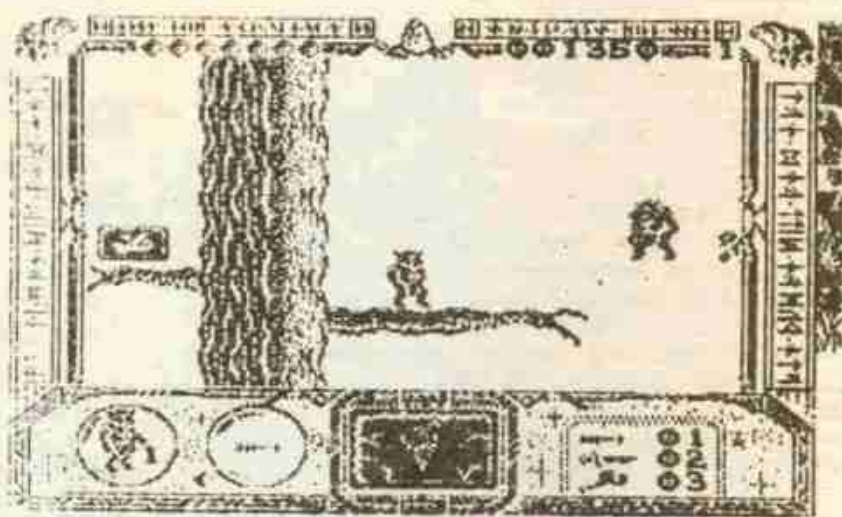
D.O.C.

Boros Péter, Győr

Az ötlet szuper, mi támogatjuk. A SYSTEM 3 telefonszámát már leírtuk. Az OCEAN címe: OCEAN Software Limited, 6 Central Street, Manchester, M2 5NS, England, az US GOLD címe pedig: US Gold Ltd., Unita 2/3 Holford Way, Holford, Birmingham, B8 7AX, England. A fejleményekről szeretnénk még hallani.

SOLDIER OF FORTUNE • Firebird

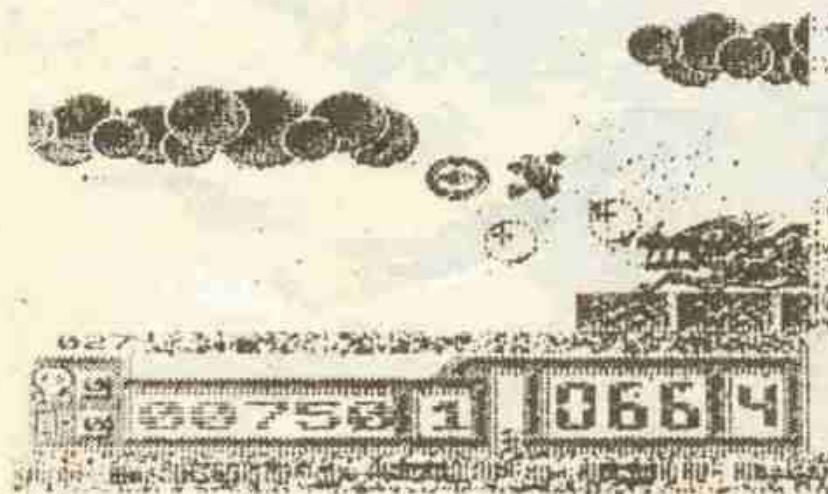
A történet főhőse, SARNAK, a Szerencse Katonája előtt nagy feladat áll: ki kell űznie az Orchard Világát előzőlő gonosz teremtményeket, ie kell győznie a legfőbb Őrt. Ez csak akkor sikerülhet, ha bejárva az egész területet megtalálja a varázspirula 6 darabját, melynek segítségével összeállítható a győzelemhez szükséges anyag. Első pillantásra úgy tűnik, a játék csupán olyan elemeket tartalmaz, amelyeket már számos más játékban láttunk: mozgó és leomló járdák, teleportáló készülékek, fák, kötelek, járatok, különböző szörnyek stb. Teendők is ismert fázisokból áll: futás, harc, tárgyak gyűjtése. A szerzők (a GRAFTGOLD csoport) azonban most is – mint az URIDIUM esetében – ismert



játéktípust valósítottak meg, igen magas szinten. A fenti elemek nagy képzelőerővel és változatossággal, nagy-szerű grafikával történő összeállítása emeli ki ezt a játékot az „egy a sok közül” kategóriából. A pályán számos választási lehetőségünk van, más és más úton érhetünk el ugyanarra a helyre. Az elrejtett tárgyakat mindig máshol találjuk meg, az ellenséges szörnyek egyre veszélyesebbek. Ellenük különböző erejű fegyvereket használhatunk – ha felleljük ezeket. No és persze érdemes az életadó kristályokat is begyűjteni.

Az akció-kaland játékok és a térképkészítés kedvelőinek bizonyosan jó szórakozást nyújt a Szerencse Katonája. A MULTIFACE tulajdonosok az örökéletet egyszerűen bevihetik: POKE 23314,0; POKE 46691,0.

NETHERWORLD • Hewson



A cím (Álvilág) nem a szervezett bűnözésre, hanem arra a pokoli környezetre utal, amelyben a következő egyszerű játékot játszunk: 10 szint mindegyikén előírt számú gyémántot kell adott idő alatt begyűjtenünk, kis korong alakú űrhajónkkal, majd egy teleportálón keresztül továbbjutunk a következő szintre. Mint látható, az alapötlet a BOULDER DASH-hez hasonlít.

Természetesen a fentiek végrehajtása távolról sem ilyen könnyű. Utunk során démonokkal találkozunk, amelyek ártalmas gömböcskéket szórnak. Persze ezeket lelőhetjük, és ez tanácsos, már csak azért is, mert az eltalált gömbök BONUS-szá változnak. Ezek közül némelyik hasznos

(életet, pontokat, feltörőket ad), mások kevésbé (energia és irányíthatóság elvesztése). Meg kell küzdenünk a mindenütt jelenlévő, gyakorlatilag elpusztíthatatlan mozgó aknákkal is. Problémát jelent a gyémántok megszerzése, hiszen ezek egyáltalán nem a pálya „nyilvánvaló” részein helyezkednek el: némelyik látszólag zárt területen, mások aknákkal védett szűk zugokban. Végül pedig a rendelkezésre álló idő hihetetlenül szoros, még akkor is, ha módunk van az óraüveg begyűjtésével 30 másodperccel visszaállítani az órát.

A NETHERWORLD jól játszható, szép grafikájú, simán scroll-ozó, gyors játék. Talán nem a HEWSON legjobb játéka, de mindenképpen megfelel a cég hírnevének.

REPTON MANIA • Alligata

A játék egyes szintjeihez szükséges password-ok:

A - nincs	E - SEASNAKE	I - ANNELID
B - ASP	F - ANEMONE	J - LEVIATHAN
C - CROCODILE	G - BASILISK	K - OPHIDIAN
D - EARTHWORM	H - CEPHALOPAD	L - KING COBRA

FERNANDEZ MUST DIE • Image Works

Hajál Fernandez-re – a játék címében szereplő diktátornak azért kell meghalnia, mert csapatai élén elfoglalta országunkat. Rajtunk a sor, hogy magányos harcosunkkal legyőzzük az Idegen sőpredéket, megszabadítsuk börtönbe vetett honfitársainkat, és leromboljuk az ellenség nyolc bázisát.

A játék **COMMANDO** típusú, de annál több stratégiai elemet tartalmaz. A függőleges scroll kissé lassú, csak akkor gyorsul fel, ha sikerül gépkocsit találnunk és beszállnunk. Az ütközések észlelése néha pontatlan: ellenséges golyók, teherautók haladnak át rajtunk anélkül, hogy meghalnánk. A háttér alakzatai (fák, barakkok, homokbuckák, vasutak, hidak) a szokásos árnyékoféssal jelennek meg.

Az ellenség leggyakoribb képviselői a gépfegyverrel folyamatosan tüzelő gyalogosok. Meg kell küzdenünk továbbá tankokkal, repülőgépekkel, és nem szabad megfeledkeznünk a számos rejtett aknáról sem. Ehhez végtelen sok gépfegyvertöltény és véges sok, de pótolható gránát áll rendelkezésünkre. Az utóbbiakkal robbanthatjuk fel például a bebörtönzött foglyok celláinak ajtaját. Lehetőségünk van egy vázlatos térkép segítségével nagyjából behatárolni pillanatnyi pozíciónkat. Végül kellemes tulajdonsága a játéknak, hogy életünk elvesztése után nem kell az egészet újra előről elkezdenünk, folytathatjuk ott, ahol befejeztük.



MIND TRAP • Mastertronic

A **MIND TRAP** egy logikai játék. A cél az, hogy egy táblán levő színes kockákat sorrendbe rakjuk. Egy kerettel 4 kockát tudunk bekeretezni és a keret mozgatásával tudjuk a kockákat mozgatni illetve jobbra-balra forgatni. A keretet csak olyan irányba mozgathatjuk, amerre köröcskék vannak a képernyőn.

Mindannak ellenére, hogy az ötlet egyszerű, a játék tökéletes, és könnyen megfertőzheti az embert. A **CRASH** szerkesztői szerint ez a program jobb mint a Tetris. Míg a Tetris tetris Indexe 82%, addig a **MIND TRAP**-é 84%. A programot egyébként egy Jugoszláv programozó készítette (Predrag Bećiric).

TIME SCANNER • Electric Dreams

Tominak hívnak? Süket vagy? Vak vagy? Röviden érzed-e úgy magad mint a "pinball wizard"? Ha minden kérdésre pozitív a válaszod, akkor ez a program, a **Time Scanner** a Te játékod. Ez a flipper valóban fantasztikus. Egyszerre 6 labdával is játszhatunk és be van építve a "TILT" is. Minden tábla két képből áll, amely nehezíti a játékot.

Everyone's A Wally • Mikro-Gen

Akik **MULTIFACE**-szel rendelkeznek az örökélet **POKE**-ot könnyen bevihetik:

POKE 58217.0: POKE 58218.0: POKE 58219.0

Az értékeket közvetlenül a játék betöltődése után kell beírniuk:

A bevittet közülük azok számára is, akik a 319/209/32768/16165 file-térképes verzióval rendelkeznek.

MERGE™ segítségével töltjük be a **BASIC loader**-t, majd módosítuk a következők szerint:

50 **LOAD** "CODE"

51 **POKE** 65345,88

52 **POKE** 65346,255

53 **FOR** I = 65368 TO 65376

54 **READ** A: **POKE** I,A: **NEXT** I

55 **FOR** I = 65400 TO 65415

56 **READ** A: **POKE** I,A: **NEXT** I

60 **RANDOMIZE** USR 55000

72 **DATA** 33,120,255,34,34,255,195,12,255

73 **DATA** 205,128,91,175,50,105,227,50,106,227,50,107,227,195,132,129

RUN, majd indítsuk a magnetofont tovább. Betöltés után végtelen élettal rendelkezünk, bármelyik szereplővel is játszunk.

WULFAN • Mastertronic

Ez egy gondolkodtató játék. A játék végére elég nehéz eljutni, még egy térkép segítségével is. Cél: hogy megtaláljuk a "rosszalkodót" és átadjuk neki bukósisakot. A labirintus tele van ajtóval, amit kulcsokkal kell kinyitnunk.

A játékban a következő tárgyak szerepelnek:

- bomba: néhány falat ledönt
- buzogány: váltásra szolgál
- üveg: ezzel lehet lefizetni Mogdun embereit, akik általában cserébe kulcsot adnak.
- pénz: váltásra szolgál
- kulcs: többféle létezik: vannak amelyek csak egy bizonyos ajtót nyitnak, vannak olyanok is, amelyek több nyitására szolgálnak.

GILBERT-ESCAPE FROM DRILL • Again-Again

Gilbert egy földönkívüli, idegen, szimpatikus zöld hős a *Get Fresh* és a *Gilbert's Fridge* játékokból. Az új játékban megbízást kapott valahol a kozmoszban. A sajátjai (*Drilliansok*) szétszedték űrhajóját és szétszórták. Főhősünknek 24 óra áll rendelkezésére, hogy összeszedje azt darabjaiból. A játék a *Pyjamarama*-hoz hasonló, jó grafikával és animációval.

Sir Fred • Mikro-Gen

A *'Palancs völgye'* így jellemezhetnénk tömören azt, amit levelirőlünk az elmúlt időszakban sorozatosan kaptunk, a már legutóbb is *'lerágott csont'*-nek nevezett *SIR FRED* örökélet bevételével kapcsolatban. Nos engedje meg a Tiszteelt Olvasó, hogy most mindent jóvátegyünk, vagyis mindhárom verzióra ismertetjük a HELYES bevétel módszert!

1. MULTIFACE törés

A SpV. 17. számának 9. oldalán található bevétel módszert szeretnénk kiegészíteni: A POKE 46650,183; III a RANDOMIZE USR 24833 (ENTER) utasítások közé szúrjuk be a következőket: ... POKE 24012,194; POKE 24013,6; POKE 24014,93; POKE 24015,0; POKE 24016,0; POKE 24017,0; POKE 24018,0; ...

2. BASIC/48963 file térképes verzió

A SpV. 6. számának 2. oldalán közölt bevétel módszer esetén a következő változtatásokra van szükség:

```
20 FOR I=65400 TO 65474: READ a: POKE I,a: NEXT I
```

```
50 DATA 5,210,0,0,245,62,183,50,58,182
```

```
60 DATA 62,194,50,204,93,62,6,50,205,93
```

```
70 DATA 62,93,50,206,93,175,50,207,93
```

```
80 DATA 50,208,93,50,209,93,50,210,93
```

```
90 DATA 241,195,68,181
```

```
100 RANDOMIZE USR 65400
```

3. FUTURESOFIT verzió

A SpV. 10. számának 9. oldalán található módszer BASIC listája helyesen:

```
10 CLEAR 65399
```

```
20 FOR I=65400 TO 65449
```

```
30 READ a: POKE I,a: NEXT I
```

```
40 DATA 49,253,255,221,33,0,91,17,0,250,62,0,56,205,86,6
```

```
50 DATA 62,183,50,58,182,62,194,60,204,93,62,6,50,205,93,62,93,50,206,93,175,50,207,93,50,208,93,50,209,93,50,210,93,201
```

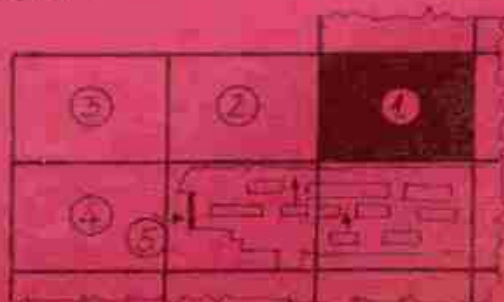
```
60 POKE 65533,68: POKE 65534,181
```

```
70 STOP
```

Ezen módosítások után nemcsak FRED-ünk lesz örökifjú, hanem a SPACE billentyű megnyomására a játék újraindul. Ez bizony kellemes dolog, mert leesni a kút mögé gyufa nélkül...

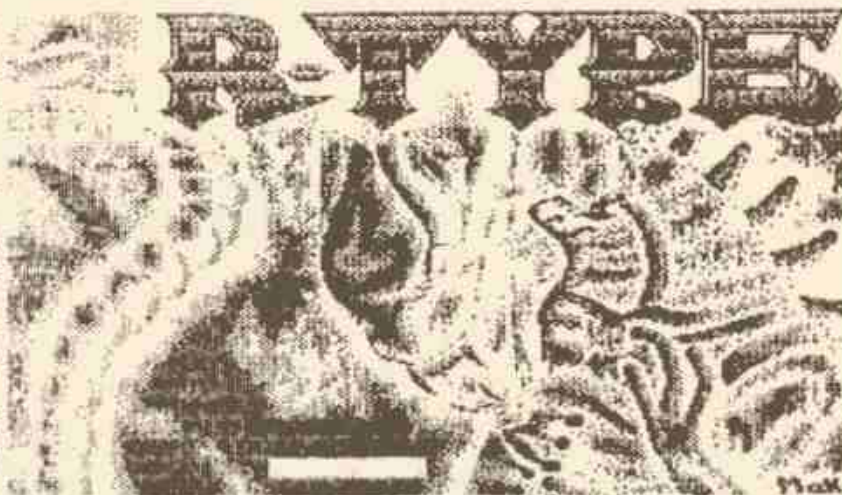
Egy kis csatás: Sokan rájöhettek, hogy a 6. nyílvesztő egy kicsit keves, azonban ha a lövést úgy adjuk le, hogy nem a gombot engedjük el, hanem a megfelelő szögrel egyszerűen átváltunk egy másik tárgyra, akkor el is röpül a nyílvesztő és meg is marad! Ez az ötlet a kőnél sajnos nem jött össze.

A 6. számban közölt térkép is hiányos volt egy helyen, ugyanis a sötét szobától (1) a varazslón (2), arkályon (3) és a királyi hátszobáján (4) át egy szekrényen keresztül (5) tükör átjárót találunk!



R-Type - Electric Dreams

Az *R-Type* az egyik legszebb, és ugyanakkor legnehezebb arcade játék. Eppen ezért úgy gondoltuk, hogy érdemes a játékot megkönnyíteni térkép, néhány útmutatás és *MULTIFACE POKE* segítségével.



- 1.szint: A nagy gyűrűt kék pontjának eltalálásával tehetjük ártalmatlanná, esetleg úgy, hogy lekapcsoljuk (DETACH) a szondát és előre küldjük a gyűrű belsejébe. A szint végén először a szörny négy szemét lövük ki, majd az előbukkanó fejet sorozzuk meg.
- 2.szint: Óvakodjunk a nagy kigyótól. A szív a tetején megjelenő kék gömb sokszori meglövésével robbantható fel, de gyorsabb, ha az űrhajónk orrára kapcsolt szondával ráereszkedve megérintjük.
- 3.szint: A hatalmas ellenséges űrhajó tetején ki-be mozgó alkatrészt kell többször eltalálnunk, vagy megérintenünk.
- 4.szint: Irsuk az állandóan növekvő zöld falat. A szint végén megjelenő űrhajó 3 darabra válik, ekkor az egyes részek zöld pontjait célozzuk meg.
- 5.szint: A rejtekéből végül előbukkanó űrhajó a középpontjára a legérzékenyebb.
- 6.szint: A nagy mozgó tömböket a középpontjukon, vagy hátulról kell többször eltalálnunk. A szint végén „csak” ki kell tartanunk.
- 7.szint: A szint végén a jobb oldalon megjelenő alakot kell bombáznunk, de az is lehetséges, hogy itt is elég bizonyos ideig élve maradnunk. Pajzsok nélkül nincs esélyünk.
- 8.szint: Ugyanaz, mint a 7.szint, de nehezebb.
- 9.szint: Aki idáig eljutott, annak már nincs szüksége útmutatásra.

Néhány hasznos, és érdekes POKE:

- 37374,0 – végtelen sok élet (bár ez önmagában nem sok segítséget jelent)
- 37362,201 – sérthetetlenség (az űrhajónak semmi sem árt)
- 38240,0
- 38241,0
- 38242,0 – űrhajónk eltűnik, így sérthetetlenné válik az ellenséges lények számára (a háttérbe való ütközés azonban most is halálos)
- 38241,6
- 38242,154 – megfordítja az űrhajót vízszintes tengelye körül
- 38241,14
- 38242,154 – megfordítja az űrhajót függőleges tengelye körül
- 38241,22
- 38242,154 – megfordítja az űrhajót mindkét tengelye körül
- 38241,254
- 38242,153 – visszaállítja az eredeti helyzetet
- 34930,195 – eltünteti a háttérrel

Végül a következő néhány byte bevitele igen hasznos kiegészítést jelent. A 'W' billentyű lenyomására az űrhajó teljes fegyverzettel rendelkezik:

CÍM	KÓD
5B00	01 FE FB ED 7B E6 02 20
5B08	08 01 10 00 11 A3 7A 21
5B10	18 5B ED 80 C3 79 89 00
5B18	01 03 01 01 01 00 08 08
5B20	00 0C 0A 01 04 09 02 06
5B28	02 00 0A 07 06 0A 0E 00
5B30	00 00 03 03 00 00 00 00

Ezután a következő két POKE-ot még vigyük be: 34388,0: 34389,91.

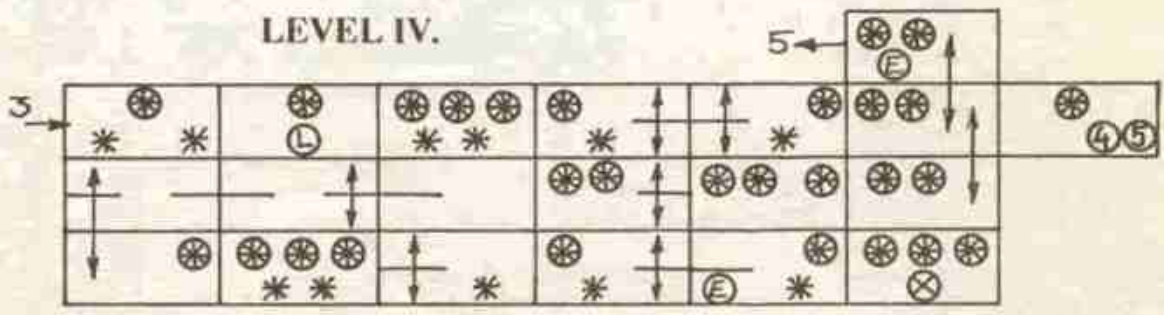
BLASTEROIDS • Mirrorsoft

Valahol a világűrben bolyong egy magányos űrhajó anyabolygója után kutatva. Útközben mindenféle ellenfelek akadályozzák. Ez a program egy tipikus lövöldözős játék.

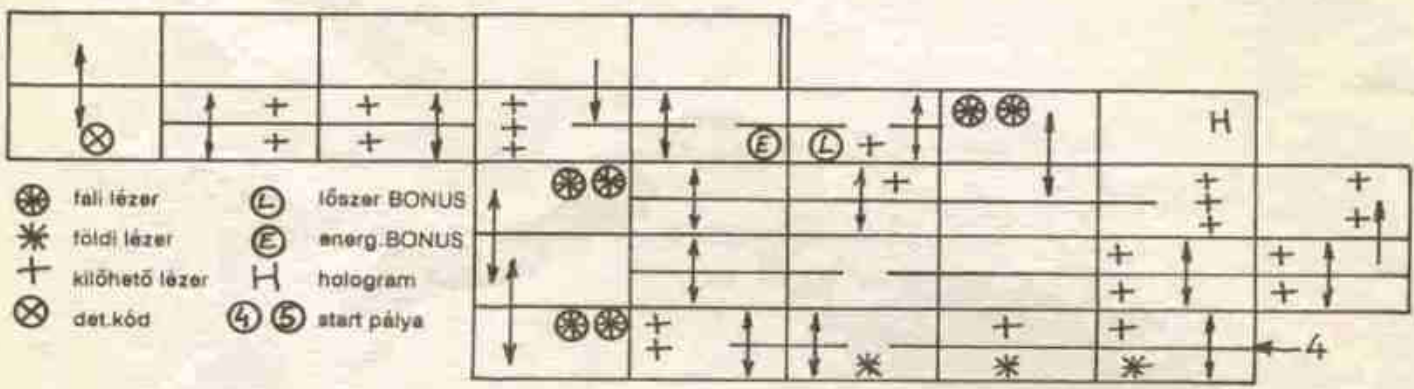
TITAN • Titus

Ez a program két ismert játék a *THROUGH THE WALL* és a *GAUNTLET* keveréke. A játék egy labirintusban játszódik ahol téglák után kutatunk. A cél az, hogy egy labdával a téglákat leszedjük. A játék közben nem kell ügyelnünk arra, hogy ha a labda elmegy mellettünk, akkor elveszítjük életünket. Amikor rátalálunk egy téglára, akkor vezessük oda a kuglit és üssük le. De ez még nem minden. Van olyan téglá is, amit nem a labdával lehet elintézni. Ezenkívül különböző zavaró ellenfelek is vannak. Vannak olyan téglák, amelyeket nehéz eltalálni, mert mozognak, a halálfejjel megjelöltek pedig veszélyesek is számunkra. Vannak kicsi és nagy felületű labirintusok. A játéknak 70-80 szintje van.

DAN DARE • Virgin (Gang of Five)



LEVEL V.



BEDLAM • GO!

Amikor meghalunk, nyomjuk meg a <C> billentyűt, akkor visszakerülünk az "élő" pozíciónkba, sőt életünk száma is maximális lesz.

AQUASQUAD • Atlantis

Amikor az üzenetek scroll-ozódnak a képernyőn, nyomjuk meg a <SYMBOL SHIFT> és a <C> billentyűket egyszerre, majd gépeljük be: 726549, s lám sérthetetlenek és örökifjúak leszünk a játékban.

KOSZMOS • Atlantis

Amikor játszunk, nyomjuk meg a <kurzor le> billentyűt, s megjelenik a menü. Most nyomjuk meg a <SYMBOL SHIFT> és a <K> billentyűket egyszerre az örökélethez.

SUPER KID • Atlantis

A főképernyőn nyomjuk meg a <G>, <D> és az <F> billentyűket, ekkor jutalomként örökéletünk lesz.

CAPTAIN FIZZ • Psyclapse

Stratégiai játék, hasonlóan a *Laser Squad*-hez. Két játékos játszhatja. A cél az, hogy az Ikarus nevű bolygóra kitelepítsünk egy fajt, az ún. *Blasfer-Tront*. Fontos, hogy a két játékos egyesítse erőit, mert csak így juthatunk el a játék végére.

CRAZY CARS 2 • Titus

Éppen hogy csak kipihentük magunkat az első résztől, amikor a TITUS programozók meghírdették a második részt. A programkészítők szerint ez a program jobb mint az előző (fékezésnél, robbanásnál, 360%-os fordulásnál). A Ferrari legújabb típusú autóját, az F40-est hajtjuk. Néha rendőrautó zavarja meg kocsinkat, lgyekezniünk kell, hogy ne érjen utól bennünket. Az úton különböző zavaró tárgyak vannak. Annak a valószínűsége, hogy egyiknek/se menjünk neki 1:1000000000000. Sok sikert...

ORIENTAL GAMES • Firebird

A *Firebird* megint valami újat készített nekünk, de az ötlet régi. A távolkeleti játékokban négyféle versengési csoport van: Kung Fu, Sumo birkózás, Kendo és az ún. hollywoodi játékok. Ha mind a négyből győztesen kerülünk ki, akkor eljutunk a az Activision legújabb GRAND versenyre, ahol elnyerhetjük a *Grand Master*-i címet.

TRANTOR THE LAST STORMTROOPER • GO!

A játékban használt kódok a következők: KEMPSTON, JOYSTICK, SPECTRUM, SOFTWARE, KEYBOARD, COMPUTER, CASSETTE, SINCLAIR, GRAPHICS, HARDWARE, TERMINAL, PRINTERS, CONTROLS, WARGAMES, WARRIORS, MEGAGAME

CHICAGO 30S • US Gold

Amikor a játék betöltődött nyomjuk meg a PAUSE gombot (<H>), majd <1> - <9>-ig a kívánt szintnek megfelelő számbillentyűt, s a kiválasztott szinten folytathatjuk a játékot.

GEMINI WING • Virgin

A szintek közötti password-ok a következők:

Level 1:	THE START	Level 3:	WHATWALL	Level 5:	SKULLDUG	Level 7:	CREEPISH
Level 2:	EYEPLANT	Level 4:	GOODNITE	Level 6:	BIGMOUTH	Level 8:	FINALFXS

LAST MISSION • US Gold

A végtelen úrhajóhoz válasszuk ki az 1. játékos-játékot, majd nyomjuk meg az (Idézőjel) billentyűt.

RED HEAT • Ocean

Nyomjuk meg a <SYMBOL SHIFT> billentyűt, valamint az összes számbillentyűt egyszerre, 10 élet és még egy-két meglepés lesz az eredménye!

HUMAN KILLING MACHINE • Capcom

Ha nem tetszik a képernyő színe, akkor nyomjuk meg a <C> billentyűt, és változtassuk meg azt!

SABOTAGE • Zepplin

A Password kódok a következők:

Level 1:	nem kell	Level 5:	ONOMASTICS5
Level 2:	BUMBLE BEE2	Level 6:	SALMAGUNDI6
Level 3:	HONORARIUM3	Level 7:	PSEUDONYMOLIS
Level 4:	PHENOMENON4	Level 8:	ONOMATOPOEIA

THE 'A' TEAM • Zafiro

A játék 2. szintjének kódja: WAEONDPEAER

MEGANOVA • Dinamic

A játék 2. szintjének kódja: 26719, a 3. szint kódja: 16640

NAVY MOVES • Dinamic

A játék 2. szintjének kódja: 83723

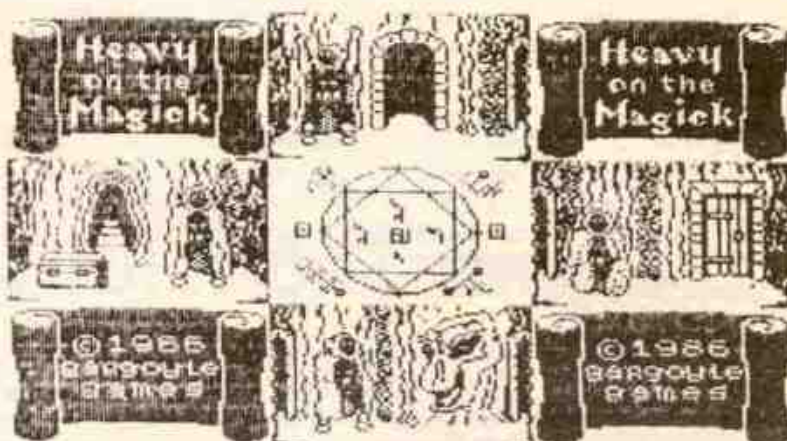
SOL NEGRO • Opera Soft

A játék 2. szintjének kódja: 2414620

A MARSPOORT megoldásának ismertetésakor (SpV. 14. szám) már ijesztgettük Olvasóinkat ama szörnyűséggel, hogy alkalomadtán sort kerítünk a GARGOYLE GAMES ezidáig utolsó arcade/adventure játékkára is. Ez a fel-emelő (?) pillanat most érkezett el: mindenki fedezékbe! – itt jön a HEAVY ON THE MAGICK. Ez a játékprogram már mindennek a teteje: kb. olyan nehéz, mint a TIR NA NOG/DUN DARACH/MARSPOORT-trilógia együttvéve, pedig azok külön-külön is elég sok kellemetlenséget okozhatnak minden vállalkozónak.

A játék kivitelezésében az említett GARGOYLE-trilógiához képest némi változás tapasztalható: a hagyományos adventure-típusú játékokhoz hasonlóan az irányítás a képernyő egy elkülönített részén szövegbejegyzéses módszerrel történik. A játék egyébként iskolapéldája lehetne az úgynevezett "interaktív" kezelésű kalandjátékoknak, mert az általunk irányított figurának kiadott utasításaink azonnal végrehajtódnak, a hatás a képernyőn is rögtön le is mérhető. A szöveges újításból kifolyólag a HEAVY ON THE MAGICK-ben fokozottabban van szükség az angol nyelv ismeretére (no meg persze egy jó adag műveltségre, intelligenciára és asszociációs mámorra). Mielőtt a Shakespeare nyelvében kevésbé járatos olvasóinkat végképp elriasztanánk a játéktól, közöljünk, hogy egy angol-magyar szótár megoldja a problémákat, sőt a játékok mindenki ismerheti, lévén megjelenésének dátuma Sinclair után 4. (műveletleneknek és C-64 tulajdonosoknak: 1986.)

Főhősünk az Axil the Able névre hallgat, foglalkozását tekintve a Búbaj Búbaj Kft. ügyvezető igazgatója. Vezetéknévéről (the Able) ítélve valamilyen feladatra rátermett férfiú – bár e feladat mibenlétét egyelőre sűrű homály fedi. Az viszont egészen bizonyos, hogy pillanatnyilag éppen egy 256 helyszínből álló, elvarázsolt barlangrendszerben rontja a levegőt, és mindenféle ördögi praktikákat követ el a barlangban tartózkodó élőlényekkel és tárgyakkal (mikor mi akad az útjába). Természetesen az élőlények ezt általában nem nézik jó szemmel (sőt, néha a "tárgyak" sem), így a móka meglehetősen életveszélyes számára.



SOME ADVICE

Talk to APEX often!

e.g. "APEX, DOOR
"APEX, WEREWOLF
"APEX, FIRE"

To dismiss say "APEX, THANKS"

Talk to other things -

sometimes they will answer!

e.g. "DOOR, password
"GUARDS, DOOR
"ASTAROT, WOLFDORP"

Finally, if in a panic,
BLAST without an object!



Betöltődés után egy kis útmutatót kapunk a játékhoz a programozóktól: eszerint gyakran kell majd beszélünk Apex-szel (kommunikációs útmutató mellékelve), valamint nem árt, ha egyéb dolgokkal is csevegést kezdeményezünk, mert néha "választ" kapunk tőlük. A "válasz" természetesen nem mindig információ, hanem néha cselekmény formájában jelentkezik. Ezután egyébként három példa is található, amely egyébként kiváló tippként szolgál a játékban azoknak, akik egyedül próbálják meg végigjátszani a játékot: Innen lehet megtudni a zárt ajtók kinyitására és a szellemek megidézésének módját. Az útmutatás végén egy elmés megállapítás található, mely szerint ha megijedünk, akkor csak robbantsunk tárgy nélkül. Örommel...

Egy billentyű megnyomása után bejelentkezik a játék főmenüje, ahol az egyes pontok előtt álló számnak megfelelő számbillentyűkkel váltogathatunk az opciók között. Aki már elkövette azt a hibát,

hogy játszott GARGOYLE-játékkal, annak nem fog sok meglepetést tartalmazni, bár néhány újdonság ebben is található:

- A **MAGICK!** választásával indíthatjuk/folytathatjuk a játékot a pillanatnyi állásból.
- **SAVE GAME** választásával kimenthetjük a játék pillanatnyi állását. Szokás szerint egy betűjelet kell megadnunk névnek. A kimentett állás a **RESTORE GAME** opcióval olvasható vissza, itt a tölteni kívánt állást jelző betűt kell megadnunk. Ha a töltés sikertelen, vagy közben megnyomjuk a **SPACE**-t, a program **ABANDONED** (feladva) felirattal visszaáll a főmenübe.
- A **SAVE/RESTORE AXIL** opciókkal a szereplőnk pillanatnyi tulajdonságait menthetjük kilöthetjük vissza. Mivel alapvetően ezek a tulajdonságok határozzák meg, hogy milyen sikerrel ténykedünk a játékban, ez egy nagyon hasznos opciópár: lehetőségünk van ugyanis arra, hogy a játékot módosított (azaz jobb) tulajdonságokkal kezdjük el. Ha tulajdonságokat töltünk be, a játék mindig előről kezdődik! – tehát nincs lehetőség arra, hogy egy bizonyos helyen sok energiát veszítve "feltuningoljuk" Axil-t. A pillanatnyi tulajdonságok (**STAMINA/SKILL/LUCK**), tapasztalati pontok (**EXPERIENCE POINTS**) és varázslói besorolás (**GRADE**) a menü jobb oldalán láthatóak – jelentőségükkel a későbbiekben még részletesen foglalkozunk.
- A **REALIGN STATUS** használatával felcserélhetjük egymással a tulajdonságok értékeit.

Az utóbbi három opció megfelelő keverésével elég jól felkészíthetjük Axil mestert az Útra. Felhívánk a figyelmet a játék egy érdekességére: ha Axil meghal (elfogy az összes életeréje), akkor a **MAGICK!** választásával – bár a játék a startszobából indul újra – az összes tárgy is visszakerül a helyére – nem előről kezdődik a játék, mert tapasztalati pontjaink, illetve az esetlegesen elért magasabb varázslói besorolás megmarad. A játék egyébként figyelembe veszi ezt a paramétert a kezdeti tulajdonságok beállításánál is. Most nézzük sorba milyen jellemzői vannak Axil barátunknak:

- STAMINA:** ez az életerőnk, ha elfogy, Axil meghal. Az életerő pontok a parancsok kiadásával (tárgy felvétele, mozgás, stb.) párhuzamosan csökken. Ugyanezt csökkentik az ellenfelek támadásai is, meghozzá annál nagyobb mértékben, minél nagyobb a támadási jellemzőjük. Egyes akadályokhoz, szellemekhez vagy ellenségekkel való érintkezés közben szintén nagyon gyorsan csökken. Növelhető viszont néhány, élelont funkcionáló "tárgy" (pl. a kenyér) felvételével illetve transzfúzió (id. később) segítségével.
- SKILL:** Axil ügyességének mértékét jelzi. Növelhető a későbbiekben használható tárgyak (pl. a Grimóiré-varázskönyv) felvételével, csökkenni csak akkor szokott, ha elvesztjük az életerőnket és új játékot kell kezdenünk.
- LUCK:** Ez a szerencsénket jelzi, legfőképpen arra van hatással, hogy a támadásaink mennyivel csökkentik az ellenfelek életeréjét, illetve azok milyen sűrűséggel támadnak vissza. Növelhető mindenféle kabalaként használható tárgy felvételével, de minden új játék kezdetekor – az előbbihez képest – 1-el csökken (ha 1 volt, akkor kisebb STAMINA-val vagy SKILL-el indulunk).

EXPERIENCE POINTS: Tapasztalati pontok. Az elnevezés magáért beszél, az eddigi játék során összegyűjtött tapasztalatoknak a kifejezése. Ezt általában más előlények kiirtásával, egyes tárgyak felvételével, illetve valamilyen más cselekménnyel növelhetjük. Ha valamilyen cselekedetünk plusz tapasztalati pontot eredményez, akkor a növekedést a program a jobb alsó sarokban lévő ablakban jelzi. A tapasztalati pontok az életerő elfogyása után is megmaradnak, és a későbbiekben – transzfúzió útján (ld.később) – életerő pontokká is átalakíthatjuk őket.

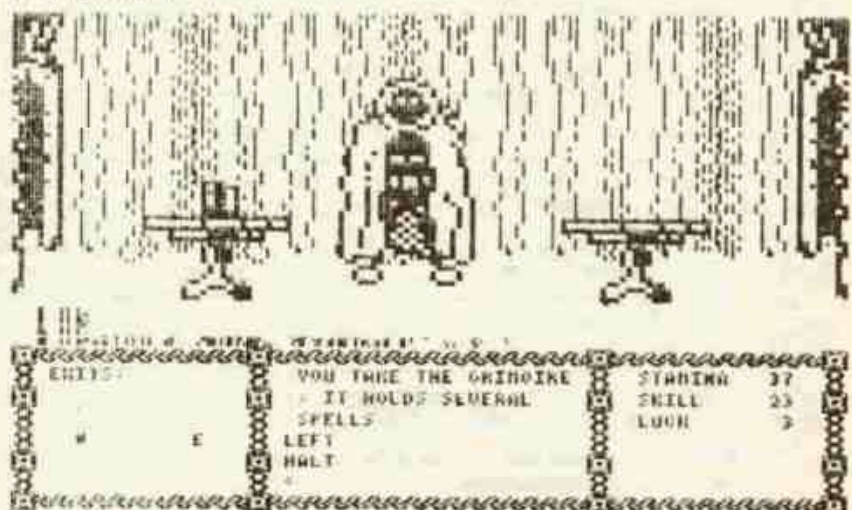
GRADE: varázslói besorolás. Ehhez a játékban egy viszonyítási szám is járul, ami talán azt jelzi, hogy Axil-nak milyen esélyei vannak a feladat végrehajtását illetően. Ennek megfelelően a kezdeti (azaz leggyengébb) NEOPYTHE besorolás viszonytszáma 1:10, míg a következő szinté (ZELATOR) 2:9, és így tovább. A varázslói besorolás a nehezebb feladatok megoldásával (pl. kulcszóval nyitható ajtók felszámolásával) növelhető, amit a program szöveggel és a képernyő villogtatásával is jelez. A tapasztalati pontokhoz hasonlóan, a rangsorolás is megmarad egy játék elvesztése után.

Az elvarázsolt barlangrendszer 4 különálló szintből áll, amelyek mindegyike 64 helyszínt tartalmaz. Az egyes szintekre a program számokkal hivatkozik, de az egyes helyszíneknek is általában saját nevük van (ezeket a térképlapon lévő térképen külön fel is tüntettük [a későbbiekben ugyanis szükség lesz rájuk], a más szintre vezető kijáratokat a szint száma jelzi). Egy játék elindítása után a barlangrendszer 2 szintjén találjuk magunkat, a **Nyomor Szobájában (ROOM OF MISERY)**. A képernyőt négy részre oszthatjuk: a felső kétharmadban látható a helyszín és az ott tartózkodó szereplők, az alsó részen pedig ablak kapott helyet, amelyek az alábbi információkat tartalmazhatják:

- Az első ablak a 'Z' billentyűvel kérhető parancsok kijelzőjeként működik, ezekkel a továbbiakban részletesen foglalkozunk.

- A középső ablakban a billentyűnyomással előállított vagy begépelte szövegeket, parancsokat illetve a játéknak az egyes eseményekre vonatkozó kommentárjait és információit láthatjuk.

- A jobb oldali ablakban láthatóak Axil bátyó pillanatnyi tulajdonságainak értékei (STAMINA/SKILL/LUCK), illetve ha valamilyen más előlény tartózkodik a szobában, akkor Axil tulajdonságai alatt megjelenik a neve, életereje (STAMINA) és ügyessége (CUNNING). Ez utóbbi egyébként arra vonatkozik, hogy ha megtámad bennünket, akkor kb. mennyivel fogja csökkenteni az életerőnket.



Mint már említettük, az irányítás az egy billentyű megnyomásával elérhető standard parancsokkal és – ha erre szükség van – annak a tárgynévnek a begépelésével történik, amire a parancs vonatkozni fog. A kurzort egy kis kör jelöli, a parancs végrehajtása a 'RETURN' megnyomásával kezdődik el. Több részből álló parancssorozat is folyamatosan végrehajtható, ha a parancsokat **vesszővel** elválasztva gépeljük be (pl. EAST,EAST,RIGHT,PICK UP BAG) vagy minden egyes parancs kiadása után megnyomjuk a 'RETURN'-t. A két módszert kombinálva is használhatjuk. Parancssorozat kiadása esetén Axil folyamatosan hajtja végre a parancsokat addig, amíg el nem fogynak vagy akadályba illetve ellenséges előlénybe nem ütközik.

Ha olyan parancsot akarunk végrehajtani, ami pillanatnyilag vagy abszolút nem lehetséges (pl. nem létező kijáraton akarunk kimenni vagy olyan tárgyat akarunk felvenni, ami nincs a szobában), Axil felénk fordul és tanácstalanul széttárja a kezét.

Bármilyen parancs(sorozat) végrehajtása azonnal megszakítható a 'H' billentyű megnyomásával elérhető HALT parancs segítségével. Ilyenkor Axil azonnal megáll és vár a további parancsokra. A HALT használatára például akkor van szükség, ha észrevesszük, hogy egy helyszínen áthaladva Axil valamilyen akadályba vagy ellenségbe ütközne (azaz vilámgyorsan elvesztené életerejét).

A mozgás az angol égtájak rövidítéseinek megfelelő billentyűkkel történik, azaz 'N'(ORTH): észak, 'E'(AST): kelet, 'S'(OUTH): dél, 'W'(EST): nyugat. Dél-nyugati (SOUTH-WEST) irányt úgy állíthatunk elő, ha az 'S' billentyű után a 'W'-t is megnyomjuk. Előfordulhat, hogy egy helyszínről nem akarunk kimenni, de valamilyen ok miatt (pl. közelgő ellenség vagy egy távolabb lévő tárgy) kénytelenek vagyunk helyet változtatni: Ilyenkor használható az 'R' illetve 'L' billentyűkkel elérhető RIGHT és LEFT parancs, amelyek hatására Axil szép komótosan átbálgal a helyszín jobb illetve bal oldalára. A LEFT/RIGHT és a HALT parancsok kombinált használatával pontosan egy adott helyre is állíthatjuk barátunkat.

Nézzük sorban a további parancsokat:

EXAMINE ('X' billentyű): Nagyon hasznos funkció, egy tárgyat vizsgálhatunk meg vele. A parancs után be kell gépeelnünk a tárgy nevét, amit meg kívánunk vizsgálni (ha ilyen tárgy nem létezik vagy a nevet rosszul adtuk meg, Axil széttárja a kezét és egy **EXAMINE WHAT?** kérdéssel érdeklődik azután, hogy ugyan mit kéne megvizsgálnia). Ha a tárgy nevét jól adtuk meg, Axil odabálgal hozzá és a középső ablakban közli az észrevételeit róla. A tárgyakat két okból is célszerű megvizsgálni:

- információkat tudhatunk meg róluk, pl. valamilyen szó van rájuk véve, ami a későbbiekben hasznos lehet számunkra, esetleg megtudjuk milyen anyagból vannak, stb. Néha ugyan Axil barátunk meglehetősen épületes megállapításokat tesz (pl. **EXAMINE SIGN** (Vizsgáld meg a jelet!) parancsra közli velünk, hogy **IT'S A SIGN** (Ez egy jel!)), de néha olyan fontos szavakat is megtalálhatunk, ami akkor nem jutna a tudomásunkra, ha csak simán felvinnénk a tárgyat.

- néhány tárgyból több is megtalálható a játékban. Ezek közül nem mindegyik az, aminek látszik, ilyenkor a vizsgálat eredményéről Axil úgy tájékoztat bennünket, hogy "úgy néz ki, mintha ... lenne" (IT LOOKS LIKE A ...). Az ilyen tárgyak **egész biztosan nem azok**, aminek látszanak: egy részük meg van mérgezve (felvételükkor egy csomó STAMINA-pontot veszünk) és általában semmire sem jók. Például rögtön kezdéskor két egyforma könyvet láthatunk a két asztalon. Vizsgálatuk után a következők derülnek ki róluk: a jobb oldali a GRIMOIRE, a varázslók kedvenc felhasználói kézikönyve, amelynek felvételével három varázslat birtokába juthatunk és 8 ponttal növelhetjük a SKILL értékét, a másik viszont csak könyvnek tűnik – meg is van mérgezve. Természetesen nem lenne ez egy igazi GARGOYLE-játék, ha ellenkező példa nem akadna: olyan dolgoknál, amelyek alapvetően káros hatással vannak ránk, ez pont fordítva van. Például arról a tárgyról, amelyik "úgy néz ki, mint egy csörgőkígyó" (IT LOOKS LIKE ROCKSNAKE), felvétele után kiderül, hogy egy vascsat (IRON CLASP), amely szerencsét hoz és növeli a tapasztalati pontokat – az igazi csörgőkígyó természetesen jól megmar bennünket, ha felvesszük. Ennyit a GARGOYLE-féle negatív logikáról, konzekvencia: mindent meg kell vizsgálni.

Ha nem tudjuk a tárgy nevét, akkor használjuk az **EXAMINE OBJECT** parancsot. Erre Axil a **hozzá legközelebb lévő** tárgyat vizsgálja meg, ami nem feltétlenül felelhet meg a tárgy jelének (lehet berendezési tárgy, jel a falon, vagy egyéb más). Ha nem a kívánt tárgyhöz ment oda a szerencsétlen, akkor a LEFT/RIGHT és a HALT parancsokkal állítsuk pontosan elé. Ha mást nem, legalább a nevét biztosan megtudjuk.

PICK UP ('P' billentyű): Tárgy felvételére szolgál. A tárgy nevét kell megadnunk utána, amelyet fel akarunk venni (ha a helyszínen nincs ilyen tárgy, vagy nem adtuk meg a nevét, Axil **PICK UP WHAT?** kérdéssel érdeklődik, hogy mit kívánunk felvenni). Ha a nevet jól adtuk meg, Axil odaballag a tárgyhöz és felveszi, amennyiben van az erszényben még hely. Összesen 6 tárgy lehet egyszerre nálunk, ha ezután is fel akarunk még venni valamit, akkor a program közli, hogy az erszény tele van (**THE POUCH IS FULL**). Nem számítanak tárgynak a felvett ételek (ezeket Axil ugyanis rogtan bekebelezi); célszerű tehát egy tárgyat letennünk, ha az erszényünk tele van, de tudjuk, hogy a közelünkben lévő tárgy biztosan étel – miután Axil elfogyasztotta a kaját, ismét felvehetjük a letett tárgyat) illetve a **GRIMOIRE** varázskönyvhöz tartozó két lap sem (**SCROLL OF PARCHMENT** – ezek beépülnek a **GRIMOIRE**-ba). Ha olyan tárgyat akarunk felvenni, amelyet nem lehet (pl. egy sziklát vagy egy oszlopot), a program közli, hogy nem tudjuk felemelni (**YOU CAN'T LIFT ...**). Egyébként ezekre nincs is szükségünk. Még egyszer felhívánk a figyelmet arra, hogy nem árt az **EXAMINE**-nal megvizsgálni azokat a tárgyakat, amelyeket fel akarunk venni! Ha a tárgy nevét nem tudjuk, itt is használhatjuk a **PICK UP OBJECT** formulát – erre Axil a hozzá legközelebb eső tárgyat próbálja meg felvenni. A **LEFT/RIGHT** és **HALT** parancsok használatával pontosan a kívánt tárgy elé állhatunk. Megjegyzendő, hogy minden egyes tárgy felvétele egy **STAMINA**-pontba kerül.

DROP ('D' billentyű): a **PICK UP** ellentéte, egy tárgyat tehetünk le vele a földre. Ha nincs nálunk ilyen tárgy vagy nem adtuk meg a nevét, Axil **DROP WHAT?** kérdéssel tudakolja, hogy mire gondolunk.

OPTIONS ('O' billentyű): Használatával visszatérhetünk a játék főmenüjébe, ahonnan kimenthetünk/visszatoíthatunk játékállást vagy Axil paramétereit – esetleg időt nyerhetünk a gondolkodáshoz. A játékot a **MAGICK**-el folytathatjuk tovább. Ha a helyszínen rajtunk kívül tartózkodik még egy élőlény vagy démon, illetve olyan szobában vagyunk, ahol varázslói szintugrás történt, nem lehet visszaugrani a menübe (**OPTIONS? NOT NOW!**) vagy át kell mennünk egy üres helyszínre, vagy meg kell ölnünk előbb az élőlényt.

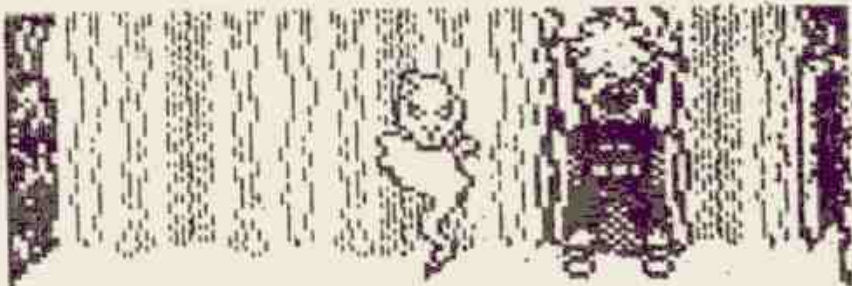
EXITS ('Z' billentyű): A 'Z' billentyű nyomogatásával négy parancsot érhetünk el, amelyek a bal oldali ablakban a helyszínre illetve Axil-ra vonatkozó információkat jelenítik meg. Az **EXITS** parancsra az ablakban az aktuális helyszínről nyíló ajtók iránya jelenik meg. A program nem tesz különbséget a pillanatnyilag használható illetve használhatatlan (pl. kulcszóval nyitható vagy egyirányú) ajtók között – **mindegyiket** jelzi. Néha az egyes kijáratoknál nyílak is megjelennek, amelyek arra utalnak, hogy az ajtón keresztül magasabb (↑) vagy alacsonyabb (↓) számozású szintre lehet átjutni. Megjegyzendő, hogy a program automatikusan ebbe az üzemmódba kapcsol, ha a helyszín felé valamilyen élőlény közeleg (**MONSTERS NEARBY**), és villogva jelzi azt az ajtót, amelyen az 2-3 másodperc múlva be fog lépni. Ilyenkor természetesen célszerű a helyszínt elhagyni vagy átbállagni a szoba másik oldalára, nehogy az összeütközéssel egy csomó **STAMINA**-pontot veszítsünk.

INVENTORY ('Z' billentyű): Leltár, az ablakban az erszényünkben lévő tárgyak neveinek listája jelenik meg. Ha valamilyen tárgyat elveszünk/leteszünk, az ablak automatikusan erre a kijelzésre kapcsol néhány másodpercre. Mint már említettük, összesen 6 tárgy lehet nálunk, többet nem tudunk felvenni, mert a program jelzi, hogy az erszény tele van (**THE POUCH IS FULL**).

MAGICK ('Z' billentyű): Azoknak a varázslatoknak a listája, amire pillanatnyilag képesek vagyunk. A varázslatokat a **GRIMOIRE** varázskönyv (**BLAST**, **INVOKE**, **FREEZE**) és a két **SCROLL OF PARCHMENT** (**CALL** illetve **TRANSFUSION**) tartalmazza. Ha nincs nálunk a **GRIMOIRE**, akkor a varázslatok listájában **NO GRIMOIRE** feliratot láthatunk – és persze varázsolni se tudunk.

SITUATION ('Z' billentyű): A helyszín nevének (**YOU ARE IN ...**), a szint számának (**ON LEVEL ...**) és varázslói besorolásunknak (**YOUR GRADE IS ...**) megjelenítése. Ha egy új helyszínre megyünk át, mindig ez kapcsolódik be.

BLAST ('B' billentyű): Robbantás varázslat, a **GRIMOIRE** felvétele után használhatjuk (ha nincs nálunk, a program a parancsra **YOU CARRY NO SPELLS** feliratot jelenít meg). A **BLAST** önmagában a helyszínen tartózkodó élőlények elleni támadásra szolgál. A robbantás az ellentél életerejét csökkenti 1-5 ponttal, a **LUCK** függvényében. Ha nincs a helyszínen ellenséges élőlény, akkor Axil szétterja a kezét és **BLAST WHAT?** feliratot láthatunk. A robbantás irányulhat egy megadott tárgyra is, ha begépeljük a parancs után a tárgy nevét – ilyenkor Axil a megadott tárgyat próbálja szétrobbantani. Vigyázat, ha a megtámadott élőlény hirtelen elhagyja a szobát, akkor az utolsó robbantást Axil önmagán fogja végrehajtani – néhány életereponti veszteséggel!



INVOKE ('I' billentyű): Szellemidézés varázslat, a **GRIMOIRE**-ban van. Az **INVOKE**-kal 4 démont idézhetünk meg a nevük begépelésével, akik különböző szolgáltatásokat tehetnek nekünk. A nevük **ASTAROT**, **BELEZBAR**, **MAGOT** és **ASMODEE**. Vigyázat, a démonok megidézéséhez egy-egy talizmánt is össze kell gyűjtenünk, másúgyonban nincs védelmünk a hatalmuk ellen (**YOU ARE NOT PROTECTED**) és bedobnak az első szinten lévő **FURNACE** nevű kemencébe, ahol tragikus hirtelenséggel el fognak fogyni az életerepontjaink. A démonidézésnek más megkötései is vannak, később még foglalkozunk a témával.

FREEZE ('F' billentyű): Fagyasztás varázslat, a **GRIMOIRE**-ban van. Meg kell utána adnunk, hogy mit akarunk fagyasztani. A képernyő ekkor villogni kezd, és a fagyasztott tárgy – a fagyasztás idejére – ugrál egy kicsit. Egy fagyasztás 4 **STAMINA**-pontba kerül. Ellentélek ellen nem hatásos (mire begépeljük az élőlény nevét, már rég megtámadott bennünket és Axil elfelejti a parancsot), de később még hasznát vehetjük...

CALL ('C' billentyű): "Hívni valakit" varázslat. Két **SCROLL OF PARCHMENT** (varázstekercs) van a játékban (vigyázat, olyan is van, ami csak úgy néz ki!), amelyeknek felvétele után kiderül, hogy a **GRIMOIRE** lapjai (bele is épülnek, tehát nem foglalnak további helyet az erszényben) és egy-egy varázslatot tartalmaznak. Az egyik tekercsen a **CALL**, a másikon a **TRANSFUSION** varázslat van. A **CALL** segítségével a többi élőlényt hívhatjuk a helyszínre a nevük megadásával. Mivel Apex kivételével mindegyik támadni fog, célszerű csak Apex-et hívogatnunk, a többitől tartózkodjunk. Apex hívására többször is szükség lehet: ő ugyanis egy jó szándékú óriás, akivel airmés csevegést folytathatunk tárgyairól, élőlényekről, ajtókról, stb. Néha fontos információhoz juttat bennünket. Ha a **CALL** varázslatot úgy akarjuk alkalmazni, hogy az azt tartalmazó tekercset meg nem találtuk meg, a program közli velünk, hogy ilyen dolog nincs (**NO SUCH THING**) – legalábbis nálunk...

TRANSFUSION ('T' billentyű): Transzfúzió varázslat, egy **CALL**-hoz hasonló tekercsen találjuk meg. A transzfúzió segítségével a tapasztalati pontokat csapolhatjuk meg, amelyek általában életerepontokká. Erre értelemszerűen akkor van szükség, ha **STAMINA**-ból gyengén állunk. Egy transzfúzió 5 tapasztalati pontot fogyaszt el, de a **STAMINA** 10 ponttal növekedik. Nem használható transzfúzió akkor, ha egy másik élőlény van a helyszínen vagy valamilyen démon bedobott minket a **FURNACE**-be.

Az elmondottakon kívül még a kommunikáció fontos része a beszéd. Beszélhetünk élőlényekkel, tárgyakkal, ajtókkal, akadályokkal – mindennel. A program beszédnek vesz minden sort, amit idézőjellel kezdünk (" azaz 'SYMBOL SHIFT + 'P'). Az idézőjel után azt kell begépelnünk, amihez/akihez a szöveget intézzük, majd vesszővel elválasztva azt, amit mondani akarunk neki, pl. "APEX, DOOR ('RETURN') begépelésére a program úgy véli, hogy Apex-hez kívánunk szólni, és az ajtóról akarunk megtudni valamit. Mint a példából is látszik csak két szóból áll a "beszéd". Hasonlóképpen kell kinyitnunk a kulcscsözelő nyitható ajtókat, pl. "DOOR, ABRACADABRA (erre ugyan egy ajtó sem fog nekünk kinyitni, de példának megteszi). Space-t nem kell használnunk a vessző után! Ha szövegünk valami hatást eredményez (esetleg válaszol valaki vagy történt valami), akkor azt a középső ablakban szöveg jelzi, ha semmi eredmény, akkor SILENCE (csend) honol a helyszínen. Előfordulhat az is, hogy olyaamihez szövegünk, ami nincs a helyszínen; ilyenkor NO SUCH THING (Nincs ilyen dolog) felirat jelenik meg. Egyébként magunkban is beszélhetünk, pl. "AXIL, AXIL – ilyenkor a program elémében megjegyzi, hogy ez a kezdődő elmebeli jele (IT'S A SIGN OF MADNESS). A csevegések általában roppant magas szellemi szinten zajlanak, az angol nyelvben kevésbé jártas játékosoknak nem árt, ha a kezük ügyében van egy angol-magyar szótár. Megjegyzendő, hogy – akárcsak a parancsoknál – a "DELETE" nem a hagyományos módon működik: megnyomásakor FORGET IT (felejtod el) felirat jelenik meg, és a program figyelmen kívül hagyja az eddig begépelte dolgokat.

Most, hogy már tisztában vagyunk a játék kezelésével és főbb szabályaival, nem árt, ha megismerkedünk a szereplőkkel. Akárcsak a többi GARGOYLE-játékban, a HEAVY ON THE MAGIC-ben is jónéhány élőlény próbálja megkeseríteni az életünket, akikbe vagy egy új helyszínre belépve botlunk bele, vagy ők látogatnak be arra a helyszínre, ahol túl sokat álldogálunk (már szó volt róla, hogy azt a bejáratot, amelyen valamilyen szörny fog némsokára belépni, a program villogva jelzi). Az élőlényeknek két tulajdonsága van: a STAMINA – akárcsak nálunk – az életerőt jelenti, amelyet sorozatos robbantással (BLAST) nullára csökkentve tudjuk a páciensét kiirtani; a CUNNING az ügyességüket jelzi, azaz hogy egy támadással (érintkezéssel) hány STAMINA-pontot csapolnak le tőlünk. Az egy Apex-et leszámítva, mindegyik élőlény roszindulatú, 2-3 másodpercenként nekünk rontanak és fogyasztják életerőnket, tehát ha találkoznak valamelyikkel, akkor vagy kezdünk el vilámgyorsan robbantgatni vagy távozzunk. A harc egyébként igen épületes látvány: a robbantásaink után egy kis füstelhő keletkezik az élőlény testén, a program közli, hogy "a robbantásnak volt egy kis hatása" (THE BLAST HAD LITTLE EFFECT), és a LUCK-pontszámunk függvényében 1-5 ponttal csökken az ellenfél életerője. Természetesen ő sem marad adósunk: időről-időre nekünk ront (... ATTACKS) és ügyességének valamint szerencsénknek alapján 1-10 életerőpontot csökkent rajtunk. Ezt Axil némi kiabálással kíséri (AAAAARGH! – fordítás talán nem szükséges), valamint elfelejti az addig begépelte parancsokat. Előfordulhat, hogy harc közben az ellenfél elhagyja a helyszínt, majd 10-20 másodperc múlva megújult erővel tér vissza. Ezzel a trükkel egyébként mi is élhetünk: ha a helyszínt elhagyjuk, majd egy kis idő múlva visszatérünk, addigra az ellenfél már általában elkötrődik onnan (hacsak nem jött közben utánunk). Ha mi győzünk (elfogynak az ellenfél életerőpontjai), a szörny eltűnik a földben (... IS DEAD) és csak egy összerombolt torzó marad belőle, valamint eredeti ügyességétől függően 1-3 ponttal növekszik a tapasztalati pontjaink száma. Ha ő győzne, a program közli, hogy szörnyű halált haltunk (YOU DIE HORRIBLY) és a játék véget ér. A győzelem záloga, hogy ne húzzunk újat a sokkal erősebb lényekkel (Werewolf-fal, Medusa-val és pláne nem Apex-szel, ő ugyanis jó szándékú), valamint a harcban minél gyorsabban robbanassunk. A különböző élőlényekkel leggyakrabban a róluk elnevezett helyszíneken találkozhatunk, pl. Troll-okkal TROLLWYND-ben, Wraith-ekkel WRAITHVALE-ben – bár ez nem törvényszerű. Az alábbi Mosolyalbum bemutatja, hogy milyen mozgó lényekkel találkozhatunk (vannak helyhez kötöttek is):



APEX

GHOST

GOBLIN

WYVERN

TROLL

WRAITH

WEREWOLF

MEDUSA

APEX THE OGRE: Nagy melák óriás, valamilyen érdemrenddel (tán R-GO jelvény?) a mellén. Ő az egyetlen jó szándékú élőlény a játékban, ő sosem támad, viszont a vele történő összeütközés jónéhány energiapontunkba kerülhet. Természetesen ha robbantunk egyet rajta, arra reagál: hirtelen letapos bennünket, lévén életerője 40, ügyessége pedig 50. Bár Apex rendkívül buta ábrázatot mondhat magának, azért elég sok sztriválója van: vele csevegve néha rendkívül értékes információkhoz juthatunk. Ha találkoznak vele, gentleman módjára mutatkoznak be neki: "APEX, AXIL "Az te vagy, te salátaagyú" – jön a válasz. Úgy látszik humoránál van, folytatásuk: "APEX, APEX "Az én vagyok" – feleli. Jó, nézzük akkor a kollégákat: "APEX, GHOST. Nem hisz a szellemekben, mert azt mondja, hogy "nincs ilyen dolog". Oké, akkor viszont mi a véleményed a Troll-okról: "APEX, TROLL. Úgy véli, hogy "egy troll-t a legjobb megölni" (tényleg!). Akkor nézzünk valami nagyobb kaliberű ellenfelet: "APEX, WEREWOLF. Azt mondja, hogy "a hagyományos módszerek a legjobbak" (tipikus GARGOYLE-féle információ!). Váltunk témát: mi a véleménye mondjuk a GRIMOIRE-ről ("APEX, GRIMOIRE). Azt mondja, mutassuk meg neki (SHOW ME THE GRIMOIRE). Ha azt mondja, hogy a kért tárgyát mutassuk meg neki (azaz tegyük le a földre), akkor majdnem biztos, hogy értéktelen információt fog mondani, például azt, hogy az bizonyosan az, ami a neve (IT'S CERTAINLY LOOKS LIKE ...), bár a GRIMOIRE letétele után azt mondja, hogy "ez egy Grimoire, néhány megjegyzéssel". Marha. Na, húzzál innen! ("APEX, THANKS) "Örülök, hogy segíthettem!" – mondja, és eltűnik.

Mint az iménti példákban is kiderült, mindenféle berendezési tárgyról, élőlényről, jelről, felvehető tárgyról el lehet beszélgetni Apex mesterrel. Bár a példákban ez okosságainak tárházából csak kevésbé részesültünk, néha kiváló segítséget szolgáltat. Kézenfekvő tehát, hogy ha valamilyen ismeretlen dologra bukkantunk, akkor hívjuk magunkhoz Apex-et és kezdünk vele beszélgetni róla. Ha a tárgy neve után azt mondja, hogy ... SOME NOTE, akkor a tárgyat valamire biztosan fel tudjuk használni! Egyébként vannak közel-foghatóbb információi is, viszont csak létező dolgokról hajlandó beszélgetni velünk, ha elvont fogalomról (pl. kulcscsözelő) vagy nem létező tárgyról beszélünk, azt mondja, hogy nincs ilyen dolog (NO SUCH THING) – bár a szellem példája azt mutatja, hogy attól még lehet. Tárgy esetében előfordulhat, hogy azt kéri, mutassuk meg neki (SHOW ME THE ...) – ilyenkor tegyük le a földre és kérdezzük ismét. Ha valamelyik tárgyról megállapítja, hogy ez bizonyosan az, aminek látszik, nem tud semmit a dologról, ne is fárasztjuk vele. Mikor megelégedtünk a beszélgetéssel, köszönjük meg a fáradozásait, mire szépen elkötrődik. Egyelőre annyit Apex bonyolult lelkivilágáról, a későbbiekben még sokszor találkozunk vele.

GHOST: Szellem, 6-os életerővel és 18-as ügyességgel. Nem ellenfél.

GOBLIN: Csimpánz-szerű, ide-oda mászkáló kreatúra. Életerejé 6, ügyessége 20, tehát őt sem túl nehéz legyőzni.

WYVERN: Ez valami sárkányfőka lehet, aki rendkívül hülyén vigyorog. Az előbbi kettőnél jóval keményebb és gyorsabb ellenfél: életerejé 10, ügyessége 25. Elpusztítása viszont nem csak 1, hanem 2 tapasztalati pontot eredményez.

TROLL: Félmisztelen púpos alak, bunkóval a kezében. Ő a leggyengébb szörny: az életerő 6, az ügyesség 15. Kevés...

WRAITH: Csontváz fekete köpenyben. Wyvern-szintű ellenfél: életerő 10, ügyesség 18. Elpusztítása 2 tapasztalati pont.

WEREWOLF: Dühösen topogó farkasember, akivel azután találkozhatunk, miután kinyitottuk a farkasok által őrzött ajtót. Rossz tulajdonsága, hogy néha cseppkőoszlopnak álcázza magát, majd hirtelen megjelenik. Ő egyébként már kemény legény, nem tanácsos ujjat húzni vele – legalábbis nem robbantással (van egy tárgy, ami megvéd tőle). Életerő 20, ügyesség 25.

MEDUSA: Egy lábastejű hölgy, akinek a bal arca pepitára van arminkeve, a fején pedig ide-oda lobogó gumiexpandereket visel. Apex-szintű nehézsúlyú szörny: életerő 40, ügyesség 50. Kár is robbantgatással próbálkozni nála, csak vesztesek lehetünk. A kalandjátékokban szereplő medúzák arról híresek, hogy pillantásukkal kővé változtatják a halandó emberfiákat. Ez a hölgy mondjuk nem csinál ilyeneket, viszont elég ronda – vajon mit szólna, ha egy tükörben meglátná magát?...

VAMPIRE: Vámpír. Életerő 40, ügyesség 50. Tulajdonságai azt mutatják, hogy nem érdemes vele ujjat húzni.

SLUG: Ronda malacpofa, pókhassal és zakóban. Életerő 50, ügyesség 35.

Az utóbbi két szereplő már nem fért be a Mosolyalbumba – mindenki megismerkedhet velük személyesen. Az utolsó négy élőlény csak bizonyos helyeken szokott előfordulni, ahova általában egy zárt ajtó kinyitásával kecmeregthetünk át. Mint a tulajdonságaik is mutatják, egyikkel sem érdemes harcba keveredni, csak akkor, ha nincs más választásunk (például elállják az egyetlen kijáratot). A harc azért is felesleges, mert mindegyik ellen védekezhetünk egy-egy kabalával: ha a hatásos kabala nálunk van, az ellenfél azonnal szörnnyethal, mielőtt nekünk ront (bár ez nem eredményez tapasztalati pontot). A kabalákról részletesen a tárgyak ismertetésénél beszélünk.

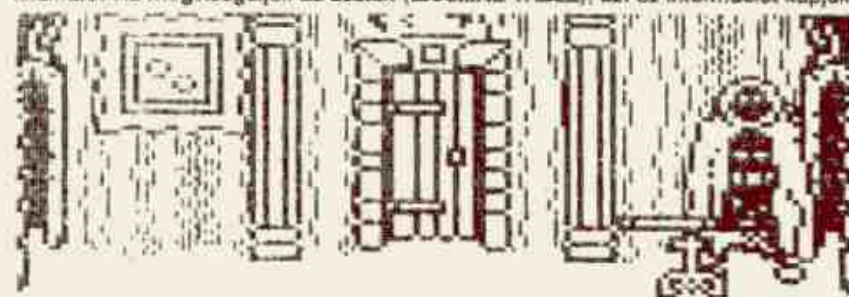
No, ezek lettek volna a szereplők. Aki a játékot ezek után egyedül kíváná végigjátszani (bár kételkedünk benne, hogy lenne ilyen mazochista), hagyja abba itt a leírás elolvasását, készítsen maga mellé egy nagy adag idegnyugtatót és vegyen ki legalább fél év fizetésnélküli szabadságot (bár lehet, hogy az kevés lesz). Célszerű előbb ismerkedni a tereppel (talán a térképmelléklet segít valamit), sok tapasztalati pontot összegyűjteni, megkeresni a GRIMOIRE másik két lapját, és sok játékállást menteni (keveset káromkodni!). Rajta ...

Mi pedig megkezdjük a megoldás ismertetését – azon a módon, amit a GARGOYLE-játékoknál eddig is követtünk: nem lépésről-lépésről (úgy 8-10 számról is elfoglalna), hanem az egyes rejtejek megoldásával illetve a tárgyak lelophelyével és funkciójával. A menetrend szabadon választott. Először is az ajtókra kerítünk sort. Egy kis mászkálás után feltűnhet, hogy néhány kijárat nem a szokásos barlangszáj alakú, hanem szabályos ajtó, amely mellett néha valamilyen jelölés is található. Természetesen zárva vannak (a térképen az átjáró áthúzással jelölve). Ha Axil-tal egy ilyen kijáratot próbálunk használni, tanácsaitanul szétárja a kezét és közli, hogy zárva van (LOCKED). A zárt ajtóknak négy fajtáját különböztethetjük meg: pénzre, kulcsszóra, kulcsra vagy egyáltalán nem nyíló ajtók. Speciális eset még a FURNACE, ahol nincs is ajtó: kijönni nem lehet, csak meghalni. Nézzük sorban a zárt ajtókat:

Számos olyan bezárt ajtó található, amely mellett egy kör alakú jel látható, benne egy fekete vonallal. Ha idehívjuk Apex-et, és megkérdezzük mi a véleménye az ajtóról ("APEX, DOOR), azt feleli, hogy kijárat Axil-nak (WAY OUT TO AXIL). Hm, hát ezzel nem sokat segítettel te nagy melák... Esetleg a jéről tudna referálni ("APEX, SIGN). Azt mondja, ez egy "nem bejárat" jelzés (IT'S A NO ENTRY SIGN). Aha, szóval ez egy inverz "behajtani tilos" tábla. Az ilyen jellel jelzett ajtókon – erről az oldalról – nem tudunk átkelni, csak valamelyik ajtó kijáratként funkcionálnak.



A következő zárt ajtó típus mellett jelleken két kört láthatunk egymás alatt. Feltűnő érdekesség, hogy az ajtók mellett egy asztalka is található. Ha megvizsgáljuk az asztalt (EXAMINE TABLE), azt az információt kapjuk, hogy ez egy asztal egy kulcs részére (IT'S A TABLE FOR A KEY).



Forduljunk segítségért ismét Apex-hez "APEX, KEY. Közli velünk, hogy mutassuk meg neki azt a kulcsot (SHOW ME THE KEY). Jaj, te szerencsétlen, mi is azt keressük! Mindegy, talán a jéről tud valamit nyilatkozni ("APEX, SIGN). Közli velünk, hogy ez egy vám jelzés (IT'S A TOLL SIGN). Aha. Mi az a vám? ("APEX, TOLL). Nincs ilyen dolog – jön a válasz. Okos vagy, Áp! Talán nem kell sok fantázia hozzá, hogy egyedül is kitaláljuk a vám a jónéhány helyen megtalálható aranyzacskók (BAG OF GOLD – vigyázat, néhánynak mérgezővel) képében testesül meg. Ha

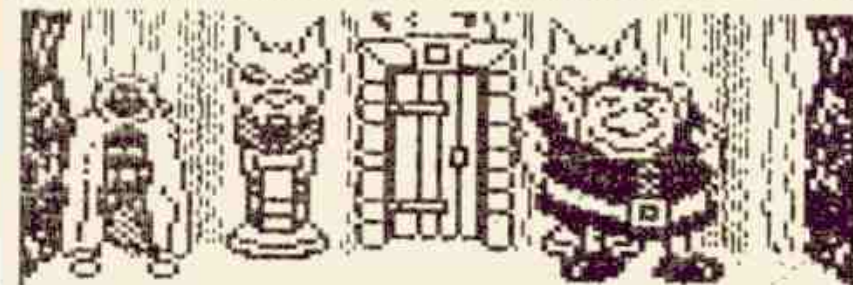
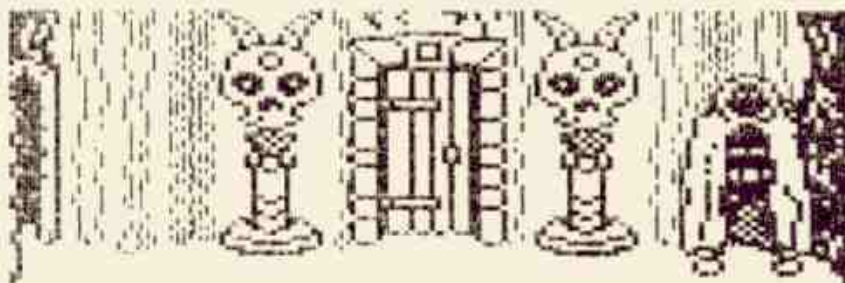
egy ilyen zacskót letesszük az asztalra (csak simán a földre nem jó!), akkor az ajtó egy CLICK! felirat kíséretében kinyílik. Természetesen ha a zacskót elveszjük az asztalról, akkor az ajtó is rögtön bezáródik. A vámos átjárók másik kijáratja természetesen nem vámos, tehát előfordulhat, hogy egy déli/déleleti/délnyugati irányú átjárón bemenne egy olyan ajtón jövünk ki, ami tulajdonképpen zárva van, innen csak vámmal nyitható. Mindenesetre nem árt, ha egy zsák aranyat mindig magunknál tartunk.

Az ajtók harmadik típusát képezik azok, amelyek egy bizonyos kulcszó kimonására (pl. "DOOR, SHAZAM) nyílnak ki. Ezeket megismerhetjük onnan is, hogy az ajtók mellett lévő oszlopok tetején valamilyen állatfejek (farkas, kecske, stb.) vannak. Ha megvizsgáljuk őket (EXAMINE OBJECT), kiderül róluk, hogy ezek őrszlopok (PILLAR OF GUARD). Szólítsuk magunkhoz Apex-et, és érdeklődünk nála, hogy mi a véleménye az ajtóról ("APEX, DOOR). Azt mondja, hogy kijárat Axil-nak. Hm, ezt egyszer már eljátszotta velünk, kár volt megint megkérdezni tőle. Akkor tudakoljuk, hogy mit tud az őrről ("APEX, GUARD). Közli velünk, hogy ilyesmi nem létezik. De hát itt vannak előttem, te ló! Tényleg, vannak – vagyis többes számban kell beszélni róluk (velük). Beszélgessünk akkor velük is egy kicsit: "GUARDS, DOOR kérdésre például valami kis segítséget nyújtanak az ajtót nyitó kulcszó megfejtéséhez. Ezek természetesen leginkább a szokásos GARGOYLE-féle segítséget jelentik, azaz legtöbbször csak akkor lesznek értelmesek számunkra, amikor már amúgy is kitaláltuk őket. A nagyobb baj azonban, hogy egy ilyen ajtó kinyitásához (tisztelőt a kivételnek) egy csomót kell mászkálnunk, egyes helyekre való eljutáshoz jónéhány tárgyat meg kell találnunk és jól felhasználnunk – és mindezek közben nem árt életben is maradnunk.

Az alábbiakban sorban ismertetjük a kulcsszavas ajtók kinyitásának módját. A kulcsszavas ajtókat a térképen A–G-ig jelöltük, a helyszínekre a hely nevével, illetve a szektorszámokkal fogunk hivatkozni. A szektorszám első száma a szint számát jelzi, a második és harmadik az X- (balról jobbra) és Y- (felülről lefelé) koordinátákat – a koordináták értelmezéséhez egyébként segítséget nyújt az 1. szint térképe melletti számozás is). Ennek megfelelően, a startszoba szektorszámja 246, míg a FURNACE a 181.

Bár a későbbiekben egyenként is ismertetjük minden egyes tárgy használatát is, az ajtók kinyitásának ismertetésénél is – ha szükséges – kitérünk néhány tárgy használatára is – további információ róluk később, a tárgyak részletes ismertetésénél).

Az A-val jelölt szoba a SECUNDA PORTA-ban van (284). Az örökkel konzultálva ("GUARDS, DOOR") tudomásunkra jut, hogy az a bizonyos szó nem is szó. Hát bizony ez egy elég épületes megállapítás... Akkor mi az ördög lehet: mondat? "DOOR, SENTENCE – semmi sem történt, azonkívül, hogy kiíródott a LOCKED (zárván) szó. Talán esetleg egy egész könyv?" "DOOR, BOOK – még mindig semmi. Talán rossz irányba tapogatózunk... Próbáljunk egy kicsit elvonatkoztatva gondolkodni a dolgokon: rájöhettünk, hogy az a szó, ami nem is szó, az talán el sem hangzik igazából. Akkor tán nem is beszéd, hanem csak valamilyen zaj. "DOOR, NOISE – semmi. Talán csak egy hang? "DOOR, SOUND – még mindig semmi. Simon és Garfunkel egy száma szerint azonban van egy olyan dolog, aminek van hangja – bár olyan mintha nem lenne. Ez pedig – a csend. "DOOR, SILENCE. CLICK! – mondja az ajtó és kinyílik. A mögötte lévő helyszínen ugyan nincs semmilyen fontos tárgy vagy élőlény, és kijárat is csak visszafelé vezet, a helyszíre belépve egy igen lényeges dolog történik: a fejünkben tomboló asszociációs mámor díjazásaképpen varázslói besorolásunk NEOPHYTE-ről ZELATOR-ra (2:9) emelkedik. Ez nagyon fontos pl. szellemidézésnél, BELEZBAR szolgáltatásait például csak ZELATOR-szintű vagy annál jobb varázsló veheti igénybe.



A B jelű ajtó WOLFDORP első szobájában van (144). Az örökkel a kulcsszó titkáról beszélgetve ("GUARDS, DOOR") azt a választ kapjuk, hogy CRY AND ENTER IT. Az AND ENTER IT jelentése nyilvánvaló: "... és lépj be". A CRY azonban egyaránt jelent sírást és kiáltást, kiabálást is. Eláruljuk, hogy a nyomorult ajtónak akármit sírhatunk – nem fog kinyílni. Tehát kiabálni kell: "DOOR, SHOUT", "DOOR, CRY" vagy "DOOR, HOWL" egyaránt hatásos – megint egy kis asszociációra van szükségünk. A kulcsszó kitalálása egyébként nem lesz túl nagy nehézség, ha figyelembe vesszük a hely elnevezését (WOLFDORP = "Farkasfalva"), az oszlopfőn lévő állatfejet meg azt, hogy ez az állat elég gyakran szokott "kiabálni", azaz ordítani. Bizony, bizony: ő a Pirokka-burger nagy hódolója – a farkas. "DOOR, WOLF – és az ajtó egy kattanással feltárol. Nem árt, ha a WOLFDORP-ban másként óvatosak vagyunk: itt ugyanis minden helyszínen várható egy kedves WEREWOLF feltűnése, aki egyébként egyike a legkellemetlenebb ellenfeleknek. Megjegyzendő, hogy előszeretettel álcázzák magukat cseppkőnek – ne csodálkozzunk tehát azon, ha a szörny közeledését jelző műszer nem jelez, mégis egyszer csak egy idegesen topogó WEREWOLF néz velünk – szó szerint – farkaszemet. A hagyományos robbantatók harci taktikáival nem nagyon tudjuk legyőzni, viszont egy ezüströg (NUGGET) nevű talizmán segít az ügyön: a farkasemberek kimondottan utálják ezt, mert mihejrt nekünk rontanak, azonnal elhaláloznak. Természetesen a WEREWOLF elleni talizmán megszerzése is némi akadályokba ütközik – erről majd később lesz szó.

A C-vel jelölt ajtó szintén WOLFDORP-ban található, csak egy ideig tekeregnünk kell addig, amíg elérünk oda (174). Az örökkel az ajtóról érdeklődve azt a választ kapjuk, hogy a belépés ide hülyeség (TO ENTER IS MADNESS). Hogyne volna az – már a HEAVY ON THE MAGICK-be is az volt... A kinyitáshoz semmi más segítséget nem találtunk – kénytelenek voltunk tehát egy kicsit (egy nagyot) gondolkodni. Végső kétségbeesésünkben megpróbáltunk a MADNESS szó szinonimáival szórakozni: ez végre célravezető megoldás volt, ugyanis kiderült, hogy a keresett szó a LUNACY.

A D jelű ajtó a QUADRA PORTA-ban van (258). Ide eljutni már egy kész Kék-túra (már amennyiben nem rendelkezünk kard (SWORD) nevű talizmánnal – ezzel ugyanis megidézhetjük ASTAROT demont, aki tudvalevőleg a teleportálás nagymestere). TROLLWYND-ből ROOM OF FLOX-ba átmenve ki kell nyitnunk az ajtót a megfelelő kulccsal; ekkor átjutunk SLYMOLE-ba, amelynek utolsó szobája a ROOM OF PURITY, ahol egy újabb kulcsos ajtó vár ránk; miután ezt is kinyitottuk, végre átkerülünk a QUADRA PORTA-ba, amelynek utolsó szobájában rálehetünk a nyomorult ajtóra. Az ajtóról tudakozódva az örök közlik velünk: THE GREAT SIGN I IN FREE. Lehet, hogy egy John Smith vagy egy Richard Davis nevet viselő úr (lévén valószínűleg angol anyanyelvűek) bizonyára messzemenő következtetéseket vonhat le ebből – nekünk azonban nagyon meggyűl a bajunk a fordítással: talán "a nagy jel én vagyok, szabadon" vagy – lévén az I éppen úgy ejtendő, mint az EYE (szem) – "a nagy jel szem szabadban"? Lehet, hogy itt kell kijutni ebből a ronda barlangrendszerből? Vagy teljesen mást jelent? Mindegy, a lényeg az, hogy kiderült, a meglehetősen számos jel közül a startszobától balra lévő helyszínen (SOTHIC COMPLEX – 236) látható dologra gondolnak. Első ránézésre úgy néz ki, mint egy SpV-keresztjejtény: se tüle, se farka. Észrevehető azonban, hogy a jelben lévő sorokat bármilyen irányban olvasva, visszafelé is ugyanazt kapjuk (Rózsa Gyuri biztos tudná rá egy speci kifejezést, de nekünk most éppen nem jut eszünkbe). Lehet próbálkozni mindentféle furtanggal, pl. lógásban összeolvasni a betűket, csigavonalban belülről kifelé és kívülről befelé, csak minden másodikat, minden harmadikat és így tovább. Mindegyik kiváló ökörségeket eredményez – de egyik sem célravezető. Egyébként ez nagyon rossz vicc volt a játék "elkövetőtől": megoldás kulcsa ugyanis innen (az ajtótól meg pláne) fénybevekre található, a H jelű ajtó mögött (427), ráadásul egy rubinba véve. A véset úgy hangzik, hogy WONTOOTOO. Ez így elég épületesen néz ki, viszont fonetikus megegyezik azzal, ha valaki angolul azt mondja: 122. Namármost, ha az 1-el "igaz"-nak



vesszük, a 2-1 pedig "hamis"-nak és a bal felső (vagy a jobb alsó – teljesen mindegy) saroktól kezdve (balról jobbra) ezen módszer alapján olvassuk össze az "igaz" karaktereket, megkapjuk az ajtó kulcsszavát. Ez talán már mindenkinek sikerülni fog – egyébként érdekes módon ez is ugyanúgy hangzik visszafelé olvasva is, mint eredetileg. A szobába belépve érdekes történés zajlik le, amire a leírás végén (most már kezd úgy tűnni, hogy az a következő SpV-ben lesz...) még visszatérünk.

Az utolsó három kulcsszavas ajtó kinyitásának módját most egy időre elnapoljuk – ezek már nagyon bonyolult cselekménysorozatok, ahol viszont feltétlenül szükség van a kulcsok által nyitható ezobák kinyitására (eddig is történt már rájuk utalás). A kulccsal nyitható ajtók pont olyanok, mint a város ajtók, csak nincs mellettük semmilyen jelzés. Az ajtók mellett egy-egy asztal található, ahol a megfelelő kulcsot elhelyezve, az ajtó kinyílik. A játékban 12 ilyen ajtót nyitó kulcsot lehetünk. A kulcsot tartalmazó helyszíneken a falon egy jel található, mindegyik egy nap jelbe beferajzolva. Megint egy kis asszociációs mámorra van szükségünk ahhoz, hogy kitaláljuk: a naposkákba rajzolt jelek a csillagövi jegyek stílusát megfelelteti – bár néha nem teljesen egyértelmű, hogy a jel mit is akar ábrázolni. Mindegy, ha sikerült kisütni, hogy mi is lehet az, meg kell keresnünk a hozzá tartozó ajtót. Az ajtó és a kulcs között a kapcsolatot a szobának a neve jelenti, ahol az ajtó van (pl. a vízöntő csillagövi jegynek megfelelő jelnél felvett kulcs nyitja az Esők Szobája helyszínen lévő ajtót). A kulcsok megvizsgálásakor kiderül róluk, hogy milyen anyagból vannak, bár ez igazából nem lényeges. Mivel – mint említettük – a kulcsokat jelölő ábrák meglehetősen stílizáltak, nem árt, ha felsoroljuk, hogy melyik ábra mit jelent:



Nyíl
CHROMA KEY



Mérleg
BRASS KEY



Halak
COPPER KEY



Ikek
LITHIC KEY



Szűz
ALUMINIUM KEY



Kos
MAGNAM KEY



Oroszlán
NICKEL KEY



Bika
IRON KEY



Bak
BRONZE KEY



Rák
TIN KEY



Skorpó
ZINC KEY



Vízöntő
COBALT KEY

CHROMA KEY: krómkulcs, WOLFDORP-ban van (121). Mivel a nyíl csillagkép tartozik hozzá, a Nyílak Szobáját (ROOM OF ARROWS) nyitja (157).

BRASS KEY: sárgaréz kulcs, ROOK OF HYDRA-ban van (346). Talán nem lesz nehéz kitalálni, hogy a Skála Szobáját (ROOM OF SCALE) nyitja (424), mivel a mérleg jelöli.

COPPER KEY: vörösréz kulcs, SOTHIC COMPLEX-ben van (255), halak csillagképpel jelölve. A ROOM OF ICHTYS-t nyitja (233), a következő összefüggés miatt: a hal az ókori rómaiak által üldözött ökeresztények vallási szimbóluma volt, görögül a neve úgy hangzik: ICHTYS. Keresztény szimbólumul azért szolgált, mert a Jesus Christus Theon Yisos Sother (Jézus Krisztus, Isten fia, Megváltó) szöveg kezdőbetűit összeolvasva ezt a szót adják. Bővebb felvilágosítás Sienkiewicz: Quo Vadis c. könyvében (HEAVY ON THE MAGICK-leírás helyett ajánljuk...).

LITHIC KEY: lítiumkulcs, WRAITHVALE-ből (261). Az ikek csillagkép jelzi – bár az ábra szerint már rögtön számi ikek – és a ROOM OF TWO ON helyszín ajtaját nyitja (327).

ALUMINIUM KEY: Alumíniumkulcs, WRAITHVALE-ben található rá (261). Meglehetősen érdekes ábra tartozik hozzá, de később kiderült, hogy a szűz csillagképre gondoltak, mivel a Tisztaság Szobáját (ROOM OF PURITY) nyitja (238).

MAGNAM KEY: Magnéziumkulcs, TROLLWYND-ben (351). Kos csillagkép és – bár fogalmunk sincs miféle összefüggés miatt (talán kizárásos alapon) – ROOM OF NANI (336) ajtaját nyitja. Esetleg volt GARGOYLE-éknak egy NANI nevű him birkájuk...

NICKEL KEY: Nikkelkulcs, SOTHIC-ban van (276). Bár az ábrából nem kimondottan lehet ráismerni, az oroszlan csillagjegyhez tartozik és ennek megfelelően a Bűszkeség Szobáját nyitja (447).

IRON KEY: Vaskulcs, METHOS-ban találjuk meg (483). A bika csillagképhez tartozik, és a Szarvak Szobáját (ROOM OF HORNS) nyitja (225).

BRONZE KEY: Bronzkulcs, GORBURG-ban botunk bele (321). A bak csillagképet találjuk mellette, ezért a Nyáj Szobáját (ROOM OF FLOX) nyitja (244).

TIN KEY: Ónkulcs, MORFANG-ban van (114). Bár jóérzésű ember a mellette lévő ábrát halaknak vagy esetleg ikeknek néző – épp ezért ez a rák. A kulcs a Karmok Szobáját (ROOM OF CLAWS) nyitja (158).

ZINC KEY: Cinkkulcs, WOLFDORP-ban található (173). Védjegye a skorpó, tehát a Fullánk Szobáját (ROOM OF STINGS) nyitja (136).

COBALT KEY: Kobaltkulcs, TROLLWYND-ben találjuk meg (364). A vízöntő jelöli, ezért az Esők Szobáját (ROOM OF RAINS) nyitja (368).

(Mindenképp őszinte rémületére közöljük, hogy a következő számban FOLYTATJUK!)

SPECTRUM programok átírása 5.



Elszántabb olvasóink okulva az előző részben leírtakból, már bizonyára betekintést nyertek a nevezetes 256 byte hosszú **LOADER** lelkivilágába. Aki még – kellő önbizalom hiányában – visszariadt ettől a lépéstől, annak megvilágítjuk a probléma hátterét. Mielőtt belemélyednénk, szeretnénk néhány tanácsot adni:

- Az **ASMON**-ban a gépi kódú programok betöltésére az "R", kimentésére az "S" parancs szolgál. Mindkettő rákérdez a start és a végcímre, valamint a betöltendő file nevére. A **SPECTRUM** programok átírásánál gondot okoz, hogy **SPECTRUM**-on a RAM terület **4000H** és **FFFFH** között helyezkedik el, míg az **ASMON** csak **801H**-tól **BFFFH**-ig tud file-okat betölteni. A probléma abból származik, hogy ilyen felállásban az abszolút címhivatkozások nem érvényesek, nehezebb megtalálni az egyes szubrutinokat. A címek megtalálásának könnyítése érdekében érdemes **4000H**-val alacsonyabb címre tölteni a programot. Ebben az esetben pl. a "CALL 7800H" utasítás által hívott szubrutin kezdőcíme esetünkben nem **7800H** lesz, hanem ennél **4000H**-val kevesebb, vagyis **3800H**. Természetesen nem ez az egyedüli és üdvözítő megoldás, viszont a későbbiekben ilyen módszerrel fogjuk ismertetni az átírás egyes fázisait.
- Az "R" parancs végrehajtása után a monitor ad egy "Last address:" üzenetet, valamint egy címet. Ezt a címet érdemes felírni, a kimentéskor ez a cím lesz a végcím.
- Ha kazettás rendszerben dolgozunk, a javított programrészleteket ne az előző változat helyére mentjük ki. Az így előálló többletkereséseket a több kazettás módszerrel védhetjük ki, nevezetesen: külön kazettán legyen a betöltő, a forrászöveg, a **SCREEN**, a program. Ez annyi kényelmetlenséget okoz, hogy kipróbáláskor oserélgetni kell a kazettákat, de ez mégis a kisebb baj. Ha lemezes rendszerünk van, akkor mindig készítsünk biztonsági másolatot, mivel a floppy az az eszköz, ahol kevés munkával igen sok adatot lehet elrontani. Ez csak az egyik ok. Előfordulhat olyan is, hogy az általunk kijavított program nem azonosul a feladatával, és utána nem is tudjuk visszajavítani a javítást. Ekkor kell visszamenni az előző, kevésbé jó, de még üzemképes változathoz.

Ennyi kis kitérő után térjünk rá a lényegre:

Töltsük be az eredeti helyére a **LOADER**-t (ez az a kivétel ami a szabályt erősíti). Az eredeti hely esetünkben **B3B0**, ez még a szabad memória területen belül van. A betöltött programot listázzuk ki ("L" parancs). Mivel tudjuk az indítási címet, érdemes itt próbálkozni (aki kökemény egyéniség, természetesen próbálkozzat máshol is, de kijelentjük, hogy igazat szóltunk).

Az indítási cím esetünkben adódik magától, mivel megegyezik a betöltési címmel. Ha elkezdjük listázni, akkor a következő látvány tárul ámuló szemünk elé:

```

B3B0 31 FF 5B LD SP,5BFF
; A verem beállítása
B3B3 3E FF LD A,FF
; Annak jelzése, hogy nem fejléc következik
B3B5 37 SCF
; A töltés jelzése (ha CY=0, akkor VERIFY)
B3B6 DD 21 00 40 LD IX,4000
; Blokk kezdőcíme
B3BA 11 00 1B LD DE,1B00
; Blokk hossza
B3BD CD E1 B3 CALL B3E1
; LOAD
B3C0 3E FF LD A,FF
B3C2 37 SCF
    
```

```

B3C3 DD 21 C4 E8 LD IX,E8C4
; A lényegét lásd fent!
B3C7 11 3C 17 LD DE,173C
B3CA CD E1 B3 CALL B3E1
B3CD 3E FF LD A,FF
B3CF 37 SCF
B3D0 DD 21 76 77 LD IX,7776
B3D4 11 A4 38 LD DE,38A4
B3D7 CD E1 B3 CALL B3E1
B3DA 31 77 E6 LD SP,E677
; A verem beállítása (ismét)
B3DD F3 DI
; Megszakítás tiltása
B3DE C3 E8 EA JP EAE8
; Indítás
B3E1 14 INC D
; Ez itt a LOAD szubrutin
    
```

Mielőtt valaki a szavahihetőségünket kétségbe vonná, sietünk leszögezni, hogy a megjegyzések, valamint a logikai egységek szétválasztása a mi merényletünk. Látható, hogy a **LOADER** 6 logikai egységből áll. Az első egy normál vagy mezei **LD SP,5BFF** utasítás. Ennek a verem állításán kívül semmi szerepe sincs. A második egység már érdekesebb. Mivel látjuk, hogy **4000H**-ra töltődik, ráadásul **1B00H** a hossza, besorolhatjuk a **SCREEN** skatulyába. A konvertálására az előző epizódban leírt játékszabályok érvényesek.

A harmadik és a negyedik egység végzi a munka orszolánrészét: ők töltik be a tulajdonképpeni programot, ráadásul két részletben. Az egyik **E8C4H**-ra **173CH**, a másik **7776H**-ra **38A4H** mennyiségű byte-ot töl.

Az ötödik egység végzi a program végleges elindítását, érdemes megfigyelni, hogy **EAE8H**-n indul a **MOONCRESTA**.

Végül a hatodik egység, a **LOAD** szubrutin, amely a kazettáról betölti az egyes file-okat. Tévedés ne essék, ez nem egy sorból áll, sőt ez a leghosszabb szubrutin a **LOADER**-ben, viszont nem láttuk értelmét teljes egészében közölni.

Akit érdekel, az tanulmányozhatja, nincs benne semmi extravagáns.

Ha valaki ennyi balszerencse és sok-sok viszály után már a hetedik menyországban érezné magát, azt ki kell ábrándítani. A cracker nagy showman lehetett, mivel még egy kis meglepetést tartogat a tarsolyában.

Töltsük be a két modult! Az elsőt **ABC4H**-ra (ez már a **4000H**-val alacsonyabb cím), a másodikat **3776H**-ra. Az első **BFFFH**-ig tart, a második **7018H**-ig, erre a kimentésnél legyünk tekintettel! Listázzuk ki a programot az indítási címtől kezdve!

```

AAE8 3E 00 LD A,00
AAEA D3 FE OUT (FE),A
; A keret (BORDER) fekete
AAEC CD C4 E8 CALL E8C4
; Ellenőrző byte a SCREEN-ből
AAEF 21 D6 E8 LD HL,E8DE
; Összehasonlítja az E8D6H
AAF2 BE CP (HL)
; memóriarekesz tartalmával
AAF3 CA 01 EB JP Z,EB01
; Ha egyezik, EB01H-ra (AB01H) ugrik
AAF6 21 48 EE LD HL,EE48
    
```

A SpV. 16. számának térkép-mellékletén már találkozhattunk ezzel a játékkal. Most megpróbálunk néhány – a játék teljesítéséhez szükséges – hasznos információt közölni, ugyanakkor a mostani térkép-mellékleten vázlatosan bejelöltük azt, hogy melyik varázslót hol találjuk meg. A szám a varázsló számát jelöli.

A SORCERY c. játékot az ENTERSOFT egyik híres programozója, Joy Broadhead dolgozta ki ENTERPRISE gépre, 1985-ben. A grafikája színvonalas, a zenéje kellemes. Itt szeretnénk azt is megjegyezni, hogy a kazettákban lévő leírás kissé hibás, ugyanis a tájékoztató szerint 3 varázslót kell kiszabadítanunk a sötét szellemidéző fogságából, a valóságban viszont 8-at!

Az 1. varázsló kiszabadítása:

A varázsló az ugyanazon pályán található könyv (*spell book*) segítségével szabadítható ki. Szálljunk rá a varázslót bezáró 'dugó'-ra, ez magától kinyílik, majd szálljunk rá a varázslóra is, és ezzel megtörténik a szabadulás.

A 2. varázsló kiszabadítása:

Ha itt vagyunk, menjünk jobbra fel, balra fel, balra fel, majd ismét jobbra fel, és már ott is vagyunk a varázslónál. Szálljunk rá az őt bezáró 'dugó'-ra, ez eltűnik, majd szálljunk rá a varázslóra...

A 3. varázsló kiszabadítása:

Ha már ezen a pályán vagyunk, akkor menjünk jobbra, jobbra, jobbra, jobbra, jobbra, jobbra le, jobbra fel, majd vegyük fel az üveget (*large bottle*), induljunk el jobbra fel, menjünk oda a serleget bezáró ajtóhoz, ez ki fog nyílni. Most vegyük fel a serleget (*goblet of wine*), menjünk vissza (balra fel, jobbra le, balra le, balra fel, balra fel, balra fel, balra le, balra le), ezután a már megszokott módon menjünk oda a varázslót bezáró 'dugó'-hoz, ami magától kinyílik, és menjünk rá a varázslóra...

A 4. varázsló kiszabadítása:

Menjünk jobbra, jobbra, jobbra le, vegyük fel a kulcsot (*door key*), majd menjünk jobbra le, jobbra fel, jobbra fel, ezután menjünk rá arra a 'dugó'-ra, amelyik egy 'holdat' (*sorcerer's moon*) zár be. Vegyük fel a holdat, menjünk balra fel, balra fel, balra le, balra le, balra le, ezt követően menjünk oda a varázsló ajtajához, ami automatikusan ki fog nyílni, s menjünk rá a varázslóra...

Az 5. varázsló kiszabadítása:

Menjünk balra, a vízésés alatt van egy címer (*coat of arms*), ezt vegyük fel, menjünk vissza a szárazföldre (vigyázzunk, nagyon könnyen belepottyantunk a vízbe, amit varázslónk nem nagyon szeret, mert minden fürdőzés alkalmával kezdhetjük újra az egészet). Ezután menjünk jobbra le (erre azért van szükség, hogy az ajtón mindig be tudjunk menni, ellenkező esetben azonban örül bezáródik utánunk), ezt követően menjünk balra, balra, balra, jobbra le, jobbra fel, jobbra, jobbra, itt vegyük fel az ékköves koronát (*jewelled crown*), majd menjünk balra, balra, balra, jobbra

le, jobbra fel, jobbra, jobbra le, menjünk a varázsló ajtajához, ami azonnal eltűnik, s menjünk a varázslóhoz...

A 6. varázsló kiszabadítása:

A bezárt varázsló felett lévő üveget (*large bottle*) vegyük fel, menjünk jobbra le, az üst alatt van egy kulcs (*door key*), menjünk neki az ajtónak, amire az kinyílik. Vegyük fel a kulcsot, majd menjünk jobbra le, jobbra fel, itt jobbra lent, a két kis üveg utáni ajtóhoz érve az eltűnik, ekkor vegyük fel a varázspálcát (*magic wand*), menjünk balra le, ezután ismét balra le, itt vigyázzunk, mert az ajtó alatt víz van, nehogy beleessünk! Ezután menjünk balra le, a varázslóhoz, nyissuk ki az ajtót, szálljunk rá a varázslóra...

A 7. varázsló kiszabadítása:

Ez csak arról a pályáról lehetséges, ahol az 1. számú varázsló volt bezárva. Ha már itt vagyunk, akkor induljunk el balra fel, balra le, itt vegyük fel a gyertyát (*fleur de lys*), majd menjünk vissza (tehát jobbra fel, ismét jobbra fel, majd középre le, ahhoz a helyhez, ahol az 1. varázsló volt). A gyertyával be tudunk menni az ajtón, itt vegyük fel a kulcsot (*door key*), menjünk rá az ajtószerrű 'dugó'-ra, ez ki fog nyílni. Battyogjunk vissza (jobbra), majd menjünk jobbra fel, jobbra le, itt vegyük fel az arany kelyhet (*golden chalice*), majd menjünk balra fel, balra fel, középre le, és menjünk be az ajtón. Menjünk le középen a nyíláson, majd lent menjünk neki jobbra az ajtónak. Most azonnal, mielőtt továbbmennénk, vegyük fel az üveget (*large bottle*), majd menjünk le jobbra, menjünk át a nyikorgó faajtón, és végül menjünk neki a varázslónak...

A 8. varázsló kiszabadítása:

Ez a szoba két részre van osztva. Ha fent vagyunk, akkor keressünk egy üveget (*large bottle*), ezzel kinyílik a 'dugó'. Ezután menjünk le. Ha már lent vagyunk, sétáljunk el jobbra le, balra fel, és vegyük fel a papírtekerccset (*scroll*). Menjünk vissza (jobbra fel, balra a középső ajtón be), csámborogjunk a beszorult varázsló tájkára, majd menjünk neki az ajtónak, ezt követően a varázslónak is...

A játék befejezése:

Menjünk jobbra le, jobbra fel, balra fel. Itt egy emelvényt látunk 9 oszloppal. Középen találjuk a legmagasabbat, mellette 4-4 oszlop mindegyikének tetején egy-egy varázsló foglalta el megérdemelt helyét. Egy dolog maradt hátra, hogy mi is elhelyezkedjünk, a még üresen maradt oszlop tetején. Ekkor...

Néhány jótanács:

- a 'dugó'-szerű ajtókat, ill. a vékony zöld ajtókat vagy a kulccsal (*door key*), vagy az üveggel (*large bottle*) tudjuk kinyitni;
- a lila 'szörnyet' egy nagy fába vágott fejszével (*sharp axe*) tudjuk megölni;
- a repülni nem tudó, földön mászkáló boszorkányt karddal (*strong sword*) tudjuk ártalmatlanná tenni;



A SORCERY igen izgalmas játék, reméljük, hogy most már könnyűszerrel teljesíthető a küldetés.

- az energilánkat az üstnél tudjuk feltölteni;
- a nem nyíló nyílorgó faajtókat a címerrel (*coat of arms*) tudjuk kinyitni;
- Két robbanószert áll rendelkezésre a 'szörnyek' totális megsemmisítésére. Az egyik egy száguldó csillagra (*shooting star*), a másik pedig egy pénzszáakra emlékeztet, \$ jellel az oldalán. Ha ezeket aktivizáljuk, ügyeljünk arra, hogy a velük való robbantáskor egyvonalban legyünk a szörnyel(-ekkel), mert csak így robban(-nak) fel.

Amaurote • Mastertronic + (?&&%?\$\$#####???)

Az ENTERPRISE-ra készült SPECTRUM program átiratokat figyelve, már nagyon profi munkákat is fellelhetünk. Ez a program is ebbe a kategóriába tartozik, ugyanis nem a "BAMBA" software ház terméke, sőt elég ha csak annyit említünk, hogy a 128K SPECTRUM hangjával zenél.

Valójában mi is a játék célja?

Az országunkat előzőnlőtték a gyilkos méhek, ezeket ki kell lrtanunk, de lehetőleg úgy, hogy a városokban minél kevesebb kárt okozunk. Ennek a feladatnak a teljesítéséhez egy lépegető terepjáró áll a rendelkezésünkre, melyet helikopterrel juttatnak el a megtisztítandó városokba. A gépünk bombákkal van felszerelve, melyekkel a méheket kell kipusztítanunk. Ez a feladat nem is olyan könnyű, mert a célzás elég nehéz a 3D felépítésű tájon. Minden – a rovarokkal történő – érintkezés növeli a sérülések mértékét (DAMAGE). A programozó bácsik gondoskodtak arról is, hogy ne unatkozzunk, mivel nem csak a dolgozókkal kell elbánnunk, hanem a jó édes mamájukat is fel kell bosszantanunk. Naná, hogy ehhez spéci bombára van szükség! Ilyet (SUPABOMB), valamint

- normál bombát (MORE BOMBS)
- menekítést a meleg helyzetekből (RESCUE)
- javítást (REPAIR)

korlátozott számban – AMIGA készlet tart – a <SHIFT> billentyűvel kérhetünk, ezen túl a kis kurzort mozgatva választhatunk a fentebb felsorolt opciók között.

Biztos feltűnt már a képernyő alján kulmináló három méh-sejt. A középső az aktuális célpontot, a két szélső a hozzánk viszonyított irányát jelöli. A célpontválasztás a következőképpen lehetséges:

- Dolgozó <Z> billentyű
- Királynő <X> billentyű
- A már rádión megkért bomba <V> billentyű
- A játéktér színe <C> billentyű

Nos, most már mindenki elmondhatja: profi AMAUROTE játékos (az más kérdés, hogy ezt ki hiszi el nekünk)!!!

Tessék szépen észrevenni a szuper grafikát, amelyet a kezdésnél láthatunk, és ez minden újratekésznél idegesítően előjön. Ja, és ha nincs örökéletes verziónk, – nekünk is csak Spectrum-ra van –, valószínűleg sokra jutunk vele. **Formalin a kivételnek!**



```

; Ha nem egyezik (váltottunk a SCREEN-en)
AAF9 01 FF FF LD BC,FFFF
; öngyilkos lesz
AAFC 11 49 EE LD DE,EE49
AAFF ED B0 LDIR
AB01 21 11 EB LD HL,EB11
; Itt folytatja, ha helyes a SCREEN
AB04 01 4A 00 LD BC,004A
; EB11H-n 4AH mnységű byte-ot XOR-ol
AB07 7A LD A,D
; az ellenőrző összeggel
AB08 AE XOR (HL)
AB09 77 LD (HL),A
AB0A 23 INC HL
AB0B 0B DEC BC
AB0C 78 LD A,B
AB0D 81 OR C
AB0E C2 07 EB JP NZ,EB07
AB11 03 INC BC
; Látszólag értelmetlen rész, ezt a
AB12 55 LD D,L
; területet módosítja az előző rutin
ABC4 21 00 40 LD HL,4000
; Az ellenőrző összeget képző szubrutin
ABC7 01 00 18 LD BC,1800
ABCA AF XOR A
ABCB AE XOR (HL)
ABCC 57 LD D,A
; D-ben az ellenőrző összeg
ABCD 23 INC HL
ABCE 0B DEC BC
ABCF 78 LD A,B
ABD0 81 OR C
ABD1 7A LD A,D
ABD2 C2 CB EB JP NZ,EBCB
ABD5 C9 RET

```

Ezek után láthatjuk, milyen gonosz volt az illető. Nézzük végig a programot. Az EBC4H-n található rutin előállít egy számot. Ezt kiszámolhatjuk az eredeti SCREEN-ből, de felesleges, elég megnézni az E8D6H címen levő byte-ot. Az első néhány sort mindjárt megspórolhatjuk. EB01H-n van az első lényeges rész, a XORolás. Mivel tudjuk az ellenőrző kódot (ha valaki nem tudná, 22H), cselesen írhatunk egy kis szubrutint, ami elvégzi helyettünk eme némes cselekedetet.

Mazochisták persze "kézzel" is megpróbálhatják. Mivel az elején található programrészre nem lesz szükségünk, ide beírhatjuk alkotásunkat. Ehhez szálljunk ki a listázásból (STOP vagy az ESC billentyű), majd nyomjuk meg az "M"-et. A módosítás címének kérésére adjuk az AAEB-at. Az ENTER lenyomása után kiíródik egy memóriadump, amiben tudunk módosítani. Szép sorban, az ENTER lenyomása nélkül írjuk be a következő számokat:

```
21 11 AB 06 4A 7E EE 22 77 23 10 F9 C9
```

Majd szálljunk ki a módosításból (STOP vagy ESC). Ha kilistázzuk ezt a kis programocskát, a következőket láthatjuk:

```

AAEB 21 11 AB LD HL,AB11
AAEB 06 4A LD B,4A
AAED 7E LD A,(HL)
AAEE EE 22 XOR 22
AAFO 77 LD (HL),A
AAF1 23 INC HL
AAF2 10 F9 DJNZ AAED
AAF4 C9 RET

```

Futtassuk le ("G" parancs, majd AAEB, ENTER), majd ha rendszeren megkaptuk a "Returned from CALL at AAEB" üzenetet, listázzuk ki az AB11H címet. Micsoda változás!

```

AB11 21 77 77 LD HL,7777
AB14 01 A4 38 LD BC,38A4
AB17 CD 47 EB CALL EB47
AB1A 21 00 EE LD HL,EE00
AB1D 01 00 12 LD BC,1200
AB20 CD 47 EB CALL EB47
AB23 21 77 77 LD HL,7777
AB26 01 A4 38 LD BC,38A4
AB29 CD 51 EB CALL EB51
AB2C 21 00 EE LD HL,EE00
AB2F 01 00 12 LD BC,1200
AB32 CD 51 EB CALL EB51
AB35 21 10 A7 LD HL,A710

```

```

; A karakterkészlet kezdőcíme
AB38 22 36 5C LD (5C36),HL
; a megfelelő változóba
AB3B 01 5E 2F LD BC,2F5E
AB3E AF XOR A
AB3F ED 42 SBC HL,BC
AB41 31 2F 75 LD SP,752F
AB44 C3 6F 00 JP 006F

```

Tanulságul álljon itt a program által használt két szubrutin is!

```

AB47 7E LD A,(HL)
AB48 ED 67 RRD
AB4A 23 INC HL
AB4B 0B DEC BC
AB4C 78 LD A,B
AB4D 81 OR C
AB4E 20 F7 JR NZ,AB47
AB50 C9 RET

AB51 7A LD A,D
AB52 AE XOR (HL)
AB53 77 LD (HL),A
AB54 23 INC HL
AB55 0B DEC BC
AB56 78 LD A,B
AB57 81 OR C
AB58 20 F7 JR NZ,AB51
AB5A C9 RET

```

Látható, hogy a program többi része itt is el van rejtve. Ismét módosítani vagyunk kénytelenek.

A módszer hasonló az előző programocskára beviteléhez, de itt az AB0F címet kell módosítanunk az alábbi byte-okra:

```

16 22 21 77 37 01 A4 38
CD 47 AB 21 00 AE 01 00
12 CD 47 AB 21 77 37 01
A4 38 CD 51 AB 21 00 AE
01 00 12 CD 51 AB C9

```

Ha ezt kilistázzuk, a következőket kell látnunk:

```

AB0F 16 22 LD D,22
AB11 21 77 37 LD HL,3777
AB14 01 A4 38 LD BC,38A4
AB17 CD 47 AB CALL AB47
AB1A 21 00 AE LD HL,AE00
AB1D 01 00 12 LD BC,1200
AB20 CD 47 AB CALL AB47
AB23 21 77 37 LD HL,3777
AB26 01 A4 38 LD BC,38A4
AB29 CD 51 AB CALL AB51
AB2C 21 00 AE LD HL,AE00
AB2F 01 00 12 LD BC,1200
AB32 CD 51 AB CALL AB51
AB35 C9 RET

```

Ha nem ezt látjuk, akkor vagy valami belerapult a szemünkbe, vagy elirtunk valamit (minden bizonnyal a Tisztelt Olvasó). Miután ezzel megvagyunk, futtassuk le ezt is ("G" AB0F, ENTER), és már meg is van a futtatható (áírható) programunk. Már csak egy akadály van hátra, nevezetesen az indítási cím. Aki azt hiszi, hogy ilyet már találtunk eleget, téved. Meg kell kerennünk a – sorrendben a harmadik – helyes belépési pontot. További torturák helyett ezt már a tudomására hozzuk azoknak a fanatikus programozóknak, akik idáig kitarítottak (jutalom): 77B2H (illetve a 4000H eltolásos módszer esetében 37B2H). Hogy az honnan származik? Ezt az Olvasóra bízunk (csak annyit segítségül, hogy A710H-2F5EH az pontosan 77B2H, valamint a SPECTRUM ROM-ban a 6FH címen egy szál utasítás van, ez pedig egy JP (HL)).

Miután így sikerült lefegyveroznünk a védelmet, akár rá is térhetnénk a program tulajdonképpeni átírására. Sajnos azonban ez a kis ujjgyakorlás sok helyet elvett, így a lényeg legközelebbre marad. Addig is, senki ne feledje el kimenteni a verejtékes munkával feltöltött programrészeket. Míg mindenki szívszorongva várja következő profeciánkat, el lehet kezdeni az ismerkedést a programozók stílusával, meg lehet próbálkozni egy ENTERPRISE LOADER készítésével.

Végezetül érdemes összefoglalni az eddigi file-ok adatait.

```

SCREEN 4000H-ra kell tölteni, 1800H hosszban.
CODE1 7776H-ra kell tölteni, 38A4H hosszban.
CODE2 EBC4H-ra kell tölteni, 173CH hosszban.

```

Ha mindent betöltöttünk, el kell ugrani a 77B2H címre.

Kellemes bogarászást!

Micro PROLOG

A PERIFÉRIÁK KEZELÉSE

A Spectrum Világ 15. számában már ismerkedtünk a Micro PROLOG fiye kezelési lehetőségeivel. Azóta sok levelet kaptunk, amelyben Olvasóink arra kértek bennünket, hogy alapszinten ismertessük a program periféria kezelési lehetőségeit, talán abból a célból, mert ez esetleg több – ott nem egyértelmű – információra adna magyarázatot.

A micro PROLOG T1.0 változata csak képernyőt, billentyűzetet, magnetofont és ZX-nyomtatót kezel, de

- ZX-nyomtató helyett eleve csatlakoztatható minden olyan készülék, amit a ROM-rutinok ZX-nyomtatóként látnak (amelyekre pl. BASIC-ben működik a COPY parancs).
- Közlünk egy egyszerű programmodosítást, mely a rendszer betöltése előtti megnyitással lehetővé teszi a #3 logikai készülék szokásos használatát (és javítjuk azt a hibát, melynek eredményeként minden sor második pozíciója ?, ha pl. programot listázunk).
- A PIO reláció segítségével a felhasználó különféle készülékeket programozhat (pl. botkormányt).
- A rendszerben elő van készítve az RS-232 csatlakozás mindkét irányú kezelése; nem túl nagy nehézségek árán ez életrekelthető.
- A microdrive periféria használatának megoldása több, de reálisan elvégezhető mennyiségű munkát igényelne (#3 természetesen irányítható a már említett módosítással is microdrive-ra).

A képernyő és a billentyűzet

A képernyőre rajzolni és írni lehet, továbbá minden, amit a rendszer vagy saját programunk billentyűzetről olvas, megjelenik a képernyőn is (ez nem vonatkozik a billentyűzet PIO relációval való vizsgálatára és a program futása közben beírt, de megjelenés előtt puffertöréssel eltüntetett adatokra).

Rajzolni a felső 11 sorba lehet, pontok és vonalak megadásával, a PNT ill. az LNE reláció kiértékelésével. HYBRID üzemmódban a szöveges információk az alsó négy sorba kerülnek (ha csak a BASIC-beli AT-nek megfelelő @V és a következő sor-ill. oszlopindex kilírása meg nem növeli az alsó rész sor-igényét), ekkor tehát a rajz első 20 sora elválik a szövegtől. NORMAL üzemmódban szöveg is a felső 22 sorba írható. Írásra a P és a PP reláció kiértékelése való, bár sok más reláció is ír a képernyőre.

Ha az utolsó szövegsor alá újabb szöveget írunk, akkor a szöveges képernyő-rész tartalma, a BASIC-ben megszokott scroll? Kérdés nélkül, eggyel feljebb lép; az első sor eltűnik, ilyenkor a közös rajz-szöveg terület (HYBRID üzemmódban a 21-22, NORMAL üzemmódban az 1-22 sor is mozog. A nyomtatás STOP (<SS+A>) megnyomásával legállítható, majd bármely gomb megnyomásával továbbindítható (ebbe a "bármely"-be nem tartoznak bele a módváltások). NORMAL üzemmódban készíthetünk szöveges rajzokat a @V vezérlő karakter segítségével (mely a BASIC-beli AT-k megvalósító CHR\$22, pontos megfelelője; hasonlóan használható @W, a TAB-t megvalósító CHR\$23 párja, de ne feledjük, hogy ez is két következő karaktert értelmez).

A rendszer aszerint van indításakor NORMAL vagy HYBRID üzemmódban, hogy közvetlenül előtte a képernyő melyik része volt outputra kijelölve. A betöltő program a rendszer indítása előtt ír a képernyő felső részére, így NORMAL állapotban indít; ha külön betöltjük a kódot és pl. RANDOMIZE USR 38762 beírásával indítjuk, akkor HYBRID állapotú képernyővel indul. Induláskor a "papír" színe sárga, a "határ" fehér, a "tinta" fekete, a képernyő nem villog és nem fényes. A felsorolt attribútumok megváltoztatására különféle relációk adnak lehetőséget. CLS, BORDER, LNE- és PNT-nél erre külön argumentumok szolgálnak; P-nél a szövegbe illesztett @P (INK), @Q (PAPER), @R (FLASH), @S (BRIGHT), @T (INVERSE) illetve @V (OVER) vezé-

lő karakterek használhatók, utánuk írva a BASIC rendszerből ismert jelentésű karaktereket. Ezek a karakterek általában szintén @ segítségével adhatók meg: @A kódja 1, @B kódja 2 és így tovább @-ig, aminek 31 a kódja. A 0 kódérték körül van némi zavar, ugyanis @@ kódja 64 (pl. (P@A)) @-t ír ki, de ugyanaz a CHAROF reláció segítségével is ellenőrizhető; a 0 kódú karakternek nincs @ segítségével megadható megfelelője (pl. ?((CHAROF X 0)(PP X)) egyetlen ?-t ír ki, mintha a 0 egy közönséges karakter kódja volna, csak éppen a képernyőre író ROM-rutin ?-t ír helyette). Egy karakter sorozatba 0 kódú karakter a STRINGOF reláció segítségével illeszthető, ha előtte CHAROF-fal 0-t adunk kód értékül egy változónak. (A 41:95 intervallumon kívüli kódú karakterek előtt álló @ hatástalan, magát @-t kivéve, mert azt csak @@-ként lehet szövegkonstansba illeszteni – ha nem párban áll, akkor összeolvad az utána következő karakterrel.) Ez a megoldás kissé körülményes, ezért a 207-CFh kódú karaktert a rendszer 0-ként írja ki (?-ként jelenik meg, ha nem vezérlő karaktert követ; a CAT-billentyűvel – <E-mód 9> – vihető be).

A vezérlő karakterek kezelésére jellemző, hogy ugyanaz a közvetlen hatásuk, mint BASIC-beli megfelelőiknek (a micro-PROLOG is a ROM rutinjait használja). Eltérés a tolvagyűrűző hatásban mutatkozik: a micro-PROLOG általában együtt állítja az ideiglenes attribútumokat az állandókkal, a micro-PROLOG (pl. BASIC-ban PRINT CHR\$18; CHR\$1; "A";PRINT"B" kiír egy villogó A-t és egy már nem villogó B-t) ?((P@R@A))(P B)) mindkét karaktert villogóként írja, sőt, villog utána minden, amit akár a rendszer, akár a program kiír, míg ennek egy olyan reláció – mondjuk CLS – kiértékelése véget nem vet, mely a villogást ki-fejezetten leállítja. Az állandó és az ideiglenes attribútumok elválasztására csak a PNT és az LNE reláció kínál lehetőséget.

P és PP között az az alapvető különbség, hogy az első változatlan formában átadja a kiírandó karakterek sorozatát a képernyőre író ROM-rutinok, míg a második minden olyan szövegkonstans idezőjel-pár közé zár, mely egyébként nem egy szövegkonstansként értelmeződne. P így lehetővé teszi a vezérlő karakterek rendeltetésszerű használatát, PP pedig az összetett szövegkonstansok tartalmának megjelenítését. PP kiértékelése képernyőn való sorváltással fejeződik be.

A kódértékük szerint a felhasználói karakterek (UDG) után következő, azaz legalább A\$H=16S kódú karakterek a képernyőre nem írathatók ki. Helyettük általában ? jelenik meg (kódjukat a ROM-rutinok hívása előtt cseréli ki a rendszer a ? kódjára, illetve CAT esetében D-ra), kivéve a DEF FN-hez tartozó, 206=CEh kódú karakter esetét, ami egyáltalán nem íródik ki (nem változik a cursor-pozíció a képernyőn), hangjelzést kapunk helyette.

A billentyűzet olvasására elsősorban az R reláció szolgál, mely kiértékelésekor az argumentumában szereplő változónak a billentyűzet-puffer következő elemi kifejezését adja értékül, ill. olvastat, ha a puffer üres, vagy a kifejezés hiányos (pl. végzárójel v. idezőjel-pár záró eleme hiányzik). K-mód nem állítható, egyébként a reláció a BASIC-ből ismert módon értelmezi a karaktereket. Kivétel a 127 kódú c karakter, mely más rendszerekben törölt jelent, olyan mintha nem is lenne (a lyukszalagos korszakban hibajavításra szolgált; a lyukcsatornás szalagon a hét értékes lyukhely mindegyikét kilyukasztva lehetett törölni egy hibás lyukkombinációt). Az R reláció is nemlétezőnek tekint, ha önmagában áll, nem egy szövegkonstans része. (Értékeltségük ki ?((R X)(PP X))-t egyszer (A c B)-t, mászor ("A" "o" "B")-t beírva R számára.) A visszaírt lista az első esetben (A B), a másodikban (A "c" B) lesz.)

Az input billentyűzetről való megadásakor – <ENTER> megnyomásáig – a beírt szöveg a BASIC-ben megszokott módon szerkeszthető ←, → és <DELETE> segítségével. Ha túl próbálunk lépni a beírt karaktersorozaton, akkor a rendszer hangjelzést ad ("kiír egy DEF FN karaktert"). A szerkesztéshez rendelke-

zére álló puffer mérete 8 képernyő-sor, azaz 256 karakter (ebből egyet a cursor foglal el). A parancs futása közben is beírható 16 karakter (a szerkesztőket is beleértve); ezeket a megkezdett input-puffer feldolgozása után (a kiírása után) veszi figyelembe a rendszer.

File-ok

A file-ok olyan, általában a számítógép periferián elhelyezkedő adatszarkazatok, melyek kezelésénél nagymértékben eltérünk az adathordozó sajátosságaitól. Az az előny, hogy ilyen – logikáinak is nevezett, magas – szinten személfeljük, mozgathatjuk adatainkat, általában bőségesen kárpótól azért a hátrányért, hogy nem használhatjuk ki a konkrét adathordozó lehetőségeit (ha nem így van, akkor lehet szükség ún. fizikai szintű periferiakezelésre).

A **micro-PROLOG T1.0** csak soros szövegfile-okat kezel, melyeket csak írhatunk és olvashatunk (file-jaiban nincsenek logikai rekordok, az esetleges blokkolást a programok elől eltakarja, nem lehet egy-egy részletet, módosítva, eredeti helyére visszairni). Néhány file a rendszerben eleve adott:

CON: író műveleteknél a képernyőre írás, olvasó műveleteknél a billentyűzetről olvasás file-ja. A képernyőkezelésénél ismertett P, PP és R reláció, definíciója szerint, egyenértékű a **CON:**-ra hivatkozó **W**, **WRITE** illetve **READ** relációval.

LST: a ZX nyomtató file-ja.

RDR: az RS232 input file-ja (pontosabban annak kezdeménye, mert érdemi része hiányzik, adatátvitelt nem végez).

PUN: az RS232 output file-ja (pontosabban annak szintén csak kezdeménye, az előbb említett ok miatt).

Ezek a file-ok mindig használhatók, a rájuk hivatkozva kiértékelte **OPEN** és **CREATE** relációk mindig sikeresek (ha nem csak inputra alkalmas fájlra adunk **CREATE**-t vagy csak outputra alkalmasra **OPEN**-t) és alapjában véve hibatlanok.

A felsoroltakon kívül a program még egy felhasználói file-tal képes dolgozni, melynek adathordozója az EAR vagy MIC jelű csatlakozónál bekötött magnetofon. E file neve minden, más célra még nem foglalt, legfeljebb nyolc karakter hosszú szövegkonstans lehet (így akár az üres is, amit másra nem is lehet lefoglalni). Nincs elvi akadálya a microdrive-kezelés megoldásának, de egy ilyen továbbfejlesztés meghaladná a munka keretait (az irodalomból nyilvánvaló, hogy a gyártó ezt a fejlesztést végrehajtotta, de nincs tudomásunk a fejlettebb program hazai előfordulásáról).

A magnetofon

Fizikai szintű kezelését a ROM rutinjai végzik – íráskor a 4C2h-n induló rutin működik, olvasáskor az 568h-n induló eleje helyett egy saját változat fut, majd a vezérlés az eredeti rutin megfelelő címére kerül (az eltérés logikailag érdektelenek). A blokkok hossza 256, bevezető bajtjuk értéke FBh. Minden blokkban az első 255 byte lehet értékes, a 256. mindig bináris 0, ezt követi a file neve – szükség esetén szóközökkel kiegészítve – nyolc karakteren, majd a blokk karakteres ábrázolású sorszáma (01 az első, 99 után 00, majd iamét 01 következik). Csak az utolsó blokkban nincs kihasználva az első 255 byte: ezt a blokkot a még hátralévő értékes karakterek vezetik be, majd utánuk a file végét jelző 26=1Ah kódú karakter áll (ha ezzel nem merül ki a lehetséges 255 byte, a folytatás akkor is érdektelen – az előzőleg kezelt blokk vége van a file-ban; kedvezőtlen esetben a file végjele egyedül is elfoglalhat egy blokkot).

A felhasználói file-ok létrehozását egy **CREATE** reláció kiértékelése vezeti be, ezt követhetik olyan relációk, amelyek írnak bele, végül egy **CLOSE** reláció teszi a file-t teljessé. Ha a **CLOSE** előtt újabb **CREATE** következik ugyanarra a file-ra, akkor készítése előről kezdődik; ha **CLOSE** előtt **OPEN** jönne, akkor az logikailag mindeneke előtt egy **CLOSE**-t hajt végre. Ha egy író reláció eredménye már nem fér a file soronlevő blokkjába, vagy **CLOSE**-t érkezik ki a rendszer, akkor képernyőre íródik a file neve és blokkjának sorszáma, majd megindul a szalagrírás. Ilyenkor a

gép kezelőjének felvételre kell kapcsolnia a magnetofont, majd – ha nem jön azonnal következő blokk – célszerű leállítania.

A felhasználói file-ok olvasását egy **OPEN** reláció kiértékelése vezeti be, ezt követhetik olyan relációk, melyek olvasnak belőle, végül egy **CLOSE** reláció jelzi, hogy a file-ra nincs tovább szükség. Ha **CLOSE** előtt újabb **OPEN** következik, előlőrl indul a file olvasása, ha pedig **CREATE** előzi meg az **OPEN**-t, akkor megkezdődik a file ismételt létrehozása (tovább nem olvasható).

Ha egy file-t nevének pontos megadásával olvasunk, akkor a rendszer az **OPEN** reláció kiértékelésekor kiírja nevét 01 sorszámmal, majd igyekszik beolvasni a file első blokkját s csak ennek sikeres megtörténte után engedi tovább a programot. Ha a következő reláció-kiértékelések végigolvasták a bentlevő blokk értékes részét és nem jutottak el a file végét jelző 26 (1Ah) kódú karakterig, akkor a rendszer ismét kiírja a file nevét, mellé a következő sorszámot, majd igyekszik beolvasni a megfelelő blokkot s csak ennek sikeres megtörténte után engedi tovább a programot. (A fizikai olvasás a blokk utolsó karakterének átadásakor zajlik le, nem akkor, mikor a következő első karakterére szükség van!)

A rendszer megengedi, hogy egy file neve az üres string (") – vagy ezzel egyenértékűen szóköz – legyen, de nem csak az ilyen file-okat olvashatjuk úgy, hogy névként üres string-et adunk meg a rájuk vonatkozó relációkban. Ha a blokkok 01-től sorban követik egymást fizikai olvasáskor, akkor mindössze annyi az eltérés ilyenkor a kezelésben, hogy **OPEN**-nél csak a 01 sorszám jelenik meg, a file-név helye üresen marad. Következő olvasáskor a sorszám eggyel nő, a file-név az először olvasott név lesz (így csak akkor nem változik, ha üres string nevű file-ból olvastunk). Ilyenkor azonban a rendszer nem ellenőrzi, hogy az olvasott blokk sorszáma megfelelő-e, hagyja feldolgozni, majd a moat már számára ismert nevű file következő blokkját várja. Ha nem jó helyen áll a szalag, akkor ebben az esetben általában hibát okoz a nem megfelelő blokk feldolgozása. A **micro-PROLOG T1.0**-nak ez a tulajdonsága lehetővé teszi, hogy az üres string nevű file-ok blokkjait tetszőleges sorrendben, egy blokkra akár többször is sort kerítve dolgoztassuk fel, csak utolsóként olyan blokkot olvassatva, mely végjelet tartalmaz (ennek neve más is lehet).

Általában ajánlható, hogy adjunk a file-oknak valódi nevet és csak akkor olvassuk őket nevük helyett üres string-et adva, ha biztosan tudjuk, hogy elsőként a 01 sorszámú blokkot fogja a rendszer megkapni (pl. szalag elején állunk, vagy előtte egy file-t végigolvastunk – a magnetofonokba épített számlálók állása ritkán elegendő biztosíték).

Ha a beolvasott blokk megfelelő, akkor az előtte kiírt név és sorszám után megjelenik a **BLOCK OK** felirat és a kiírás hely ugyanennek a sornak az eleje lesz. Így sorozatos olvasásnál ugyan ide kerül a következő blokk neve és sorszáma, letörölve az előző **BLOCK OK** feliratot. **CLOSE** kiértékelése sort vált a képernyőn, így a folyamat minimális számú sor-leptetéssel jár. Ez a helytakarékos megoldás zavart okozhat olyankor, ha nem sorozatban olvassuk a blokkokat, hanem közben használjuk a képernyőt más célra is. Hogy a billentyűzött input nem üres sorban jelenik meg a képernyőn, az csak némi többlet-figyelmet igényel, de hogy egy esetleges hibajelzés számát jobbról kiegészítse a blokk sorszám második jegye, az komoly táány lehet.

Ha a beolvasott blokk nem az, amit a rendszer várt, akkor kiírja a nevet és sorszámát – ugyanoda, ahova egyébként **BLOCK OK**-t – , majd olvassa a következő blokkot. Ha olvasási hiba van (más a bevezető byte értéke, rövidebb a blokk vagy nem egyezik a hosszanti paritás), akkor ugyanott **READ ERROR** felirat jelenik meg és szintén a következő blokk olvasása következik (a kezelőnek megfelelő helyre kell tekernie a szalagot).

A nyomtató

A rendszer nyomtató-kezelése **ZX**-nyomtatót tételez fel, melyre ugyanaz és ugyanúgy írható, mint képernyőre (eltékvintve olyan természetes különbségektől, hogy a nyomtató nem tudja visszahúzni a papírt és nem szines). Legegyszerűbben úgy nyomtathatunk, hogy a TO billentyű (<SS+F>) megnyomásával bekapcsoljuk a "másolat" (hardcopy) funkciót. Ennek az az eredménye,

hogy minden, amit a rendszer a CON: file-ra ír, kikerül az LST: file-ra is. E funkció TO ismételt megnyomásával állítható le. Fontos megjegyezni, hogy a CON-ról érkező input nem kerül át LST:ra, így a készülő lista nem dokumentálja a végzett munkát, nem "igazi hardcopy".

A nyomtató használatának szokásos módja az LST: file-ra való írás. Erre elsősorban a W és a WRITE reláció kiértékelése szolgál, bár más relációk is írhatók fájlba, így írhatunk nyomtatóra is. Rajz nyomtatására a rendszer nem ad lehetőséget, a képernyőre készült rajzok sem másoltathatók segítségével nyomtatóra (bár az utóbbi gépi kódú megoldása minden konkrét, "rajzolni tudó" nyomtatóra egyszerű feladat).

Ha olyan nyomtatóval dolgozunk, melyet a ROM-rutinok ZX-nomtatóként érzékelnek, akkor csupán a vezérlő karakterekkel kell óvatosan bánnunk; minden ugyanúgy használható, mint BASIC-ban.

Ha másképp szeretnénk nyomtatni — pl. RS232 felhasználásával vagy microdrive-ra irányítva a nyomtatandókat —, akkor beleutközünk a rendszer egy viszonylag egyszerűen áthidalható korlátjába: a micro-PROLOG T1.0 mindig a #3. logikai csatorna rutinjaival kezelteti a képernyőt, a billentyűzetet és a nyomtatót, ahelyett hogy az utóbbi esetben a 3. logikai csatornához rendelt rutinjal dolgozna. Hogy ezt tegye, a következő módosítások eleveendőek:

929Ch-tól 3 bájtra írjuk a következőt: CDh, 6Fh, 97h;

976Fh-tól 17 bájtra pedig a következőt: 2Ah, 4Fh, 5Ch, FDh, CBh, 01h, 4Eh, C8h, D6h, EDh, 5Bh, 1Ch, 5Ch, 2Bh, 19h, D1h, C9h.

Ha a gépi kódú program indítása előtt #3-t a megfelelő fizikai csatornához rendeljük, akkor a nyomtatás eredménye a hozzárendelt készülékre kerül (microdrive esetén néhány dologra ügyelni kell: egyszer a program előtt nincs hely két microdrive-csatorna számára, ezért ha a programot is onnan töltjük be, akkor #3-t a betöltés után nyissuk meg; másszor nincs a rendszerben mód #3 lezárására, ezért némi fölösleges írással gondoskodjunk arról, hogy az értékes információ vége is az adathordozóra kerüljön, ahonnan BASIC-ban MOVE segítségével vehető elő; harmadszor hiba esetén — pl. ha megtelik a kazetta — a vezérlés kikerülhet a interpreterből és nincs garancia arra, hogy munkánk eredménye nem vesz el, a RANDOMIZE USR 32750 valószínűleg segít).

RS232 esetén, ha a nyomtatót vezérelni akarjuk, használjuk a B csatornát.

Bármilyen nyomtatónk van, találkoznunk néhány könnyen javítható hibával. Egy rossz ugrás miatt bizonyos karakterek másképp kerülnek nyomtatóra, mint képernyőre — elsősorban a CAT nem használható 0 kódú karakter kilírására (ami a vezérlést nehezíti). Javítása: írjunk 91EEh-ra 94h-t.

LISTP feltételezi, hogy a kilíró rutinok megőrzik az A regiszter értékét. A CON:ra és a felhasználói fájlokra író rutin valóban megőrzi, a többi azonban nem. A nem működő PUN: érdektelen, míg életre nem keltek, LST: esetében megfelel a következő javítás:

735Dh-tól 2 bájtra írjuk a következőt: 80h, 97h.

9780h-tól 6 bájtra pedig a következőt: F5h, CDh, 76h, 91h, F1h, C9h.

Ezáltal az eredeti, regisztertartalom-ronító rutin-hívást betesszük egy mentés-visszatöltés pár közé és így a nyomtatott listákon is jó lesz a sorok eleje.

KILLED UNTIL DEAD

LEVEL III. - CASES FOR THE CUNNING

1. pálya

(The Case of the Mutilated Moose)

gyilkos: Peter

áldozat: Sydney

hely: Patio

fegyver: gun

ok: He ran over your brother

Break In

Sydney - —

Peter - Bina Crossby (3.)

Claudia - —

Agatha - Bare Knuckle Boxing

Mike - Qua Seraisera

2. pálya

(The Mystery of the Leaping Fish)

gyilkos: Claudia

áldozat: Peter

hely: hall

fegyver: gun

ok: Peter stole your "fish" plot

Break In

Sydney - Traris McGee

Peter - Soff soled shoes

Claudia - Sherlock

Holmes

Agatha - Murp the Surf

Mike - Cricket

3. pálya

(Paint by Numbers)

gyilkos: Claudia

áldozat: Peter

hely: hall

fegyver: gun

ok: Peter ruined your reputation

Break In

Sydney - Vincenzo

Agatha - —

Peter - Mary Tyler Moore

Mike - —

Claudia - Blackjack

4. pálya

(Practical Pastimes)

gyilkos: Mike

áldozat: Agatha

hely: library

fegyver: gun

ok: She pulled too many practical joker

Break In

Sydney - Dr. Antuirete

Louis

Peter - Sir Alic Guinness

Claudia - Baker Street

Agatha - Beekeeping

Mike - Maigret

5. pálya

(A Stich in Time)

gyilkos: Mike

áldozat: Claudia

hely: Mike's room

fegyver: knife

ok: Claudia squeezed you out of the deal

Break In

Sydney - A circus

elephant

Peter - Jimmy Hotta

Claudia - Chicago

Agatha - Yankee

Mike - Billy the Kid

Desolator

CHEAT ÖTLET

A játék MULTILOAD-os, azaz az egyes szintek csak az előzők teljesítése után tölthetők be. Ezt átverhetjük, ha egy előző pálya fejébe után egy következő pálya fő-bodját gépeljük be. Meglátjuk, így a további szintek is játszhatók!

HISOFT 'C' COMPILER

A programszerkesztő használata

A Spectrum C rendszerének használatát a programszerkesztő ismerettségével folytatjuk, majd ezután térünk rá magának a nyelvnek az ismertetésére. Mint említettük a szerkesztőbe az EDIT majd az ENTER billentyűmegnyomásával léphetünk be. A szerkesztő minden számmal kezdődő sort beszerkeszt a pufferébe, hasonlóan a BASIC-hez. A sorszám nem lesz része a programnak, csak a szerkesztéshez szükséges. A szerkesztőnek egy-karakteres parancsai vannak, melyek a következők:

I <sorszám1>, <sorszám2> – automatikus sorszámadás. Az első szám az első sornak a száma, míg a második szám a növekmény. A szerkesztő addig adja a megfelelő sorszámokat, míg be nem viszünk egy sort, amelyik csak az <EDIT> kódját (kis téglalap) tartalmazza.

L <sorszám1>, <sorszám2> – listázás az első számtól a második számig.

K <sorszám> – beállítja, hogy hány soronként történjen a listázás.

W <sorszám1>, <sorszám2> – ugyanaz, mint az L, csak a nyomtatóra listáz.

V – Kírja az aktuális elválasztójelet és az utóljára használt <sorszám1>, <sorszám2>, <param1>, illetve <param2> értékeket, továbbá a szövegfile kezdő és végcímét.

S, <param1> – a parancsok paramétereit közt általában a vessző áll. Ha ez valamiért nem megfelelő, akkor átcsereíthatjuk a <param1> sztring első karakterére. Ezt nem célszerű szököznie, vagy ENTER-nek választani!

C – Visszatérés a fordítóba.

B – Kilépés a BASIC-be. Újraindítás a RANDOMIZE USR 25200 parancsral!

N <sorszám1>, <sorszám2> – a sorok átszámozása. Az első megadott szám az első sor száma, míg a második a növekmény.

F <sorszám1>, <sorszám2>, <param1>, <param2> – a megadott sorok közt (beleértve a határokat) keresel a <param1> sztringet, s ha megtalálta, áttér szerkesztő módba (lásd később). Lehetőségünk van a soron belül további előfordulást keresni, vagy a megtalálást helyettesíteni a <param2> sztringgel. A sztringeket nem szabad idézőjelek közé tenni!

P <sorszám1>, <sorszám2>, <param1> – a puffer megadott sorait kírja a szalagra <param1> néven.

G, <param1> – beolvassa a <param1> nevű file tartalmát a pufferbe.

Ez utóbbi két parancs esetén, ha a sztring második karaktere kettőspont, akkor az előtte álló szám a microdrive sorszám, a név pedig a harmadik karaktertől kezdődik.

Utoljára hagytuk az **E** <sorszám1> szerkesztési parancs ismeretét. Ennek segítségével a létező, <sorszám1> sorszámú sort tudjuk megszerkeszteni. Ehhez az alábbi alparancsokat használhatjuk:

<Kurzor jobbra> – egy karaktert átmásol a régi sorból az újba.
<Kurzor balra> – viszerakja az átmásolt karaktert a régi sorba.
<ENTER> – a szerkesztés befejezése, az új sor bekerül a pufferbe.

Q – a szerkesztés vége, az eredeti sor megmarad.

R – az eredeti sor szerkesztését előlről kezd.

K – törli a kurzor után álló karaktereket.

Z – a kurzor alatti karakterrel együtt törli a sor végéig.

I – beszúrás. Villogó * kurzor jelenik meg. A felesleges karaktereket a <DELETE> gombbal törölhetjük. A beszúrást befejezhetjük az <ENTER> megnyomásával.

X – az összes maradék karaktert átmásolja, a sor végére áll, s áttér beszúrási módba.

C – helyettesítési mód. A kurzor + lesz, s a bevitt karakterek átírják az eredeti sorban levő karaktereket. Kilépni az <ENTER> megnyomásával tudunk.

F <param1>, <param2> – befejezi a sor szerkesztését, kicseréli a régit és megkeresi a következő sort, ahol az F után beírt sztring megtalálható.

S – az F parancsban megadott sztringre cseréli a megtalált sztringet, majd tovább keres.

A szerkesztő parancsai és az E parancs alparancsai úgy működnek, hogyha valamelyik paramétereiket (ezek a <sorszám> és <param> értékek) nem adjuk meg, akkor az utóljára megadott értékekkel dolgozik. Ezeket lehet a V parancsral lekérdezni.

Tekintettel arra, hogy a C program képes önmaga felülírására, minden programfuttatás előtt célszerű a 10 sornál hosszabb

programokat eimenteni! Javításnál nem kell mindig, csak akkor, ha már nem emlékszünk, mit is javítottunk ki.

A C nyelv általános tulajdonságai

Egy C program változó és típusdeklarációk valamint függvények halmaza. A függvényeken belül már nincs lehetőség további függvények deklarálására; míg a függvényen belül használhatunk további változódeklarációkat. Ennek következtében a változókat külső és belső változóknak hívjuk, attól függően, hogy az összes függvényen kívül, vagy valamelyiken belül deklaráltuk. (Más nyelvek esetén a globális-lokális elnevezéspár használatos.) A belső változók mag – deklarációjuktól független – lehetnek statikusak vagy sem. A statikus változó értéke a függvényből való kilépéskor is megmarad és a függvény újabb hívásakor ezzel az értékkel számolhatunk tovább. Valamennyi használt változót – az első használata előtt – deklarálni kell.

A C nyelv maga lehetővé teszi extern és register típusú változók használatát is. Ez utóbbi azt fejezi ki, hogy a változót magát, ha ez lehetséges, a processzor regiszterében valósítjuk meg. Erre a Spectrum esetében lehetőség, a Z80 regiszterei mögött már foglaltak. Az extern típus azzal van összefüggésben, hogy a C nyelven írt program egyes darabjait külön file-ban is tárolhatjuk, külön fordíthatjuk, ilyenkor jelezni kell a file elején, hogy melyek azok a függvények és változók, amelyek deklarációját egy másik file tartalmazza. Erre nekünk nincs lehetőségünk, mert a fordító mindig a forrásfile-ból és egyetlen menetben állítja elő a futtatható kódot.

A C erősen típusos nyelv: minden változónak van típusa, s ezt a program szövegében expliciten meg is kell adni. A C aránylag kevés típust ismer, ezek a következők:

C elnevezés	Jelentés	Tárolási hossz
1. char	karakter	1 byte
2. int	egész szám	2 byte
3. short	egész szám	2 byte
4. long	egész szám	2 byte
5. float	valós szám	nem implementált
6. double	dupla pontos	nem implementált

Általában a C megvalósításokban az int, short és long típusok nem azonosak, de ebben a reprezentációban igen. Mint a bevezetésben már említettük, a valós számokat nem kezeli a rendszer. A 2-4. típusok elé tehetünk egy unsigned módosító szócskát, ilyenkor az illető változók nem tárolhatnak negatív számot.

A változó-deklarációk alakja igen egyszerű:

[static] <típus megnevezése> <változó lista> ;

Például:

```
int i, j;
char beolv, kír, o;
short alfa, beta;
static unsigned ciklusvalt;
```

mind érvényes deklarációk. Ha a static jelzőt nem tesszük ki, akkor a változó nem lesz statikus, amit néha úgyis ki szoktunk fejezni, hogy a változó automatikus. (Természetesen az alaptípusokon kívül vannak ún. típusképző operációk – pl. tömbök – amelyek segítségével további típusokat lehet definiálni.)

Az alaptípusokon – elsősorban a számokon – a megszokottnál lényegesen több műveletet lehet végezni, s azok jelölése is meglepő. Ezek a műveletek (és relációk) a következők:

C jelölés	elnevezés
1. !	logikai tagadás
2. ~	bitenkénti komplementer-képzés
3. +	összeadás
4. -	kivonás, ellentett képzés
5. *	szorzás, mutatóképzés
6. /	osztás
7. %	maradékképzés
8. <<	balra tolás
9. >>	jobbra tolás
10. <	kisebb
11. >	nagyobb
12. <=	kisebb egyenlő
13. >=	nagyobb egyenlő
14. =	egyenlő
15. !=	nem egyenlő
16. &	bitenkénti és
17.	bitenkénti vagy
18. &&	bitenkénti kizáró vagy
19.	logikai vagy
20. .	sorozatos kiértékelés
21. ? :	feltételes kifejezés
22. ++	növelés
23. --	csökkentés

24.	=	1	értékadás
25.	+=	1	értékadás összeadással
26.	-=	1	értékadás kivonással
27.	*=	1	értékadás szorzással
28.	/=	1	értékadás osztással
29.	%=	1	értékadás maradékképzéssel
30.	>>=	1	értékadás jobbrtolással
31.	<<=	1	értékadás balrtoállással
32.	&=	1	értékadás bitenkénti és-sel
33.	=	1	értékadás bitenkénti vagy-gyal
34.	^=	1	értékadás bitenkénti kizáró vagy-gyal
35.	sizeof		változó tárolási hossza
36.	cast(< típus >)		konvertálás

Az egyoperandusú műveletek prioritása a legmagasabb (nem számítva a zárójeleket). Ezek a következők: 1 , $*$, $\&$, $-$, $!$, $-$, $++$, $--$, $sizeof$, $cast$. Ezek az operátorok jobbról balra kötnek. Az összes kétoperandusú művelet balról jobbra köt. Ezek prioritását a táblázatban soroltuk fel. Ezek után következnek prioritásban az értékadó műveletek, amelyek mindegyike jobbról balra köt. Végül legalacsonyabb prioritású a vessző művelet, s így mindig ez hajtódik végre utoljára. A vessző balról jobbra csoportosít.

A C nyelvben az értékadás mint önálló utasítás nem jelenik meg, hanem speciális műveletként szerepel. Az $x=2$ értékadás eredményül 2 ad, s mellesleg az az érték az x változóba is belemásolódik. Pl. az $y+(x=2)$ művelet korrekt. Eredményül az $y+2$ értéket kapjuk, s mellékhatásként az x változóba kerül a 2 érték.

Hasonlóan furcsa a vessző művelet, bár a HISOFT-ban nem implementálták. Mégis megemlítjük, mert a C programokban igen gyakran használják. Pusztán annyit jelent, hogy a vesszővel elválasztott kifejezéseket sorban ki kell értékelni, s magának a kifejezésnek az értéke az utoljára kiértékelte kifejezés lesz. Tekintsük például az $i=(j=2)+1$ kifejezést! Először természetesen a zárójelen belüli kifejezés értékelődik ki. Ennek hatására j értéke 2 lesz, majd kiértékelődik a $j+1$ kifejezés, melynek értéke így 3 lesz. Ezért i a 3 értéket kapja, s magának az egész kifejezésnek az értéke is 3 lesz!

Következő furcsaság a $++$ típusú értékadások. A C nyelv készítői úgy találták, hogy igen gyakoriak az $X=X+1$ típusú értékadások, s ezek gyors végrehajtása érdekében bevezették a $++$ értékadást. $X++=Y$ az $X=X+Y$ értékadás rövidítése. Ha tehát az X változót 1 -gyel akarjuk növelni, akkor a leggyorsabban azt az $X++=1$ paranccsal hajthatjuk végre. Hasonlóan kell értelmezni az összes többi értékadó műveletet.

Az 1 -gyel való csökkentést és növelést annyira lényegesnek találták, hogy külön csökkentő és növelő műveleti jelet iktattak a nyelvbe, ezek a $++$ és a $--$ jelek. Ezeket csak változó elé vagy után lehet írni. Ha elé írjuk, akkor a változó értékét először kell módosítani, majd utána használni. Ha utána írjuk, akkor előbb használjuk a változó értékét, s csak utána módosítja a program. Például az $i=(x=2,y=3,x+++(++y))$ művelet hatására i értéke 6 lesz! (Vajon miért?)

Utoljára maradt a feltételes értékadás. Ez egy háromargumentumú művelet: $<feltétel> ? <kifejezés> : <kifejezés>$. A feltételes kifejezés értéke a $>$ követő első kifejezés, ha a feltétel igaz, különben a második. Például az $x>y ? x : y$ kifejezés értéke mindig az x és y számok közül a nagyobbik.

C programok felépítése

Mint említettük, a C program változódeklarációk és függvénydefiníciók sorozata. A program futása a main nevű függvény meghívásával kezdődik, ez tekinthető tulajdonképpen a főprogramnak. Egy függvénydefiníció az alábbi alakú:

```
[[extern] <típus>] <név> (<paraméterlista>)
<változó-deklarációs lista 1>
{ <változó-deklarációs lista 2>
  <utasítási lista> }
```

A $<név>$ a függvény nevét adja meg, ezzel a névvel kell a függvényre majd hivatkozni. A paraméterlista a függvény formális paramétereit adja meg, ezeket és csak ezeket a $<változó-deklarációs lista 1>$ -ben deklarálni kell. Ezeket a változókat csak és kizárólag a függvény belsejében használhatjuk. A kapcsos zárójelek közti részt szokás függvénytorzának hívni. Ez ugyancsak tartalmazhat egy deklarációs listát, ezek lesznek a függvény belső változói. Végül a függvénytorzában kell megadni a végrehajtandó utasítások sorozatát. A függvény nevét egy típusazonosító előzheti meg, ez megadja, hogy a függvény milyen típusú értéket tér vissza. Ha elmarad, akkor az mindig int típusú jelent. Ennek jelzésére, hogy a függvény visszaadott értéke érdektelen szokás a void típust használni. A void típus valójában egész típust jelent, de a függvény nevét elé írásával egyértelműen jelezhetjük, hogy nincs felhasználható visszaadott érték. Az extern használatára még a későbbiekben visszatérünk.

A továbbiakban felsoroljuk az összes C utasítás formáját. Ügyeljünk az egyes utasítások végén látható pontosvesszőre (;) használatuk kötelező!

1. Összetett utasítás. Az összetett utasítás alakja:

```
{ <változó-deklaráció lista > <utasítási lista > }
```

alakú, hasonló a függvénytorzssal. Az itt szereplő változók csak az összetett utasításon belül léteznek, onnan való kilépéskor értékek nem használható fel.

2. A $<kifejezés>$; szintén utasítás. Hatására a $<kifejezés>$ kiértékelődik, s értéke rendelkezésre áll. Ha a kifejezésben értékadások is szerepeltek, akkor a kifejezés kiértékelése egyben az értékadások végrehajtását is jelenti.

3. Függvényből visszatérni kétféleképpen lehet. Vagy a függvény torzsenek utolsó utasítását is végrehajtuk, vagy kiadjuk a return utasítást. Ennek alakja kétféle:

```
return ; vagy return <kifejezés > ;
```

Ez utóbbi esetben a függvény értéke a $<kifejezés>$ lesz, míg az előző esetben nem definiált. Az első (tehát a $<kifejezés>$ nélküli) esetet akkor használjuk, ha nem akarunk a függvényvel értéket visszaadni.

4. Vezérlésátadások. Bár a C strukturált nyelv, lehetőség van címkek használatára a programban, s a megcímkézett utasításra való közvetlen vezérlésátadásra. Erre szolgál az utasítások címkével való ellátása, illetve a goto utasítás. Ezek alakja a következő:

```
goto <címke > illetve <címke > : <utasítás >
```

A címke tetszőleges, másra még nem használt azonosító lehet. A goto utasítás végrehajtása azt eredményezi, hogy a vezérlés a $<címke>$ -vel megjelölt utasítással folytatódik. A címke csak az éppen végrehajtás alatt álló függvényben lehet.

A feltételes vezérlésátadásra a

```
if (<kifejezés >) <utasítás > illetve a
```

```
if (<kifejezés >) <utasítás 1 > else <utasítás 2 >
```

utasítások szolgálnak. A kifejezés kiértékelése után a vezérlés az $<utasítás 1 >$ feldolgozásával folytatódik - feltéve, hogy a kifejezés értéke igaz volt, pontosabban nullától különböző. Ha a kifejezés értéke hamis, azaz nulla, akkor az első esetben a következő utasításra kerül a vezérlés, míg a második esetben az $<utasítás 2 >$ kerül végrehajtásra. A C tartalmaz egy többszörös elágaztatást is lehetővé tevő utasítást. Ennek alakja a következő:

```
switch (<kifejezés >) <utasítás >
```

míg az $<utasítás >$ -ban az alábbi utasítások fordulhatnak elő:

```
case <állandó-kifejezés > : <utasítás > illetve default :
```

```
<utasítás >
```

A switch utasítás végrehajtása a $<kifejezés>$ kiértékelésével kezdődik. Ezután a program megkeresi az utasításban az első olyan case -t, amelyikhez tartozó állandó kifejezés értéke megegyezik a $<kifejezés>$ éppen aktuális értékével. Az ezt a case -t követő első utasítással folytatódik a program. Ha ilyen case nincs, akkor a default-ra kerül a vezérlés. Ha a default nem szerepel a switch -en belül, akkor a switch követő első utasításra kerül a vezérlés. Szemben a C nyelv definíciójával, a HISOFT C-ben a switch utasításon belül már nem definiálhatunk semmit.

5. Ciklusutasítások. A C háromféle ciklusutasítást ismer. Ezek a következők:

```
while (<kifejezés >) <utasítás >
```

```
do <utasítás > (<kifejezés >);
```

```
for (<kifejezés 1 >; <kifejezés 2 >; <kifejezés 3 >)
```

```
<utasítás >
```

A két while -t tartalmazó struktúra hasonló. Az $<utasítás >$ rész újra és újra végrehajtódik egészen addig, amíg a kifejezés igaz, azaz nem nulla. Amikor a $<kifejezés>$ értéke először lesz nulla, a ciklus lejár. A do -val kezdődő ciklus annyiban tér el, hogy az $<utasítás >$ legalább egyszer végrehajtódik.

A for utasítás egyenértékű a következő while ciklussal:

```
<kifejezés 1 >;
```

```
while <kifejezés 2 > {
```

```
<utasítás >
```

```
<kifejezés 3 >
```

```
}
```

Ennek megfelelően az első kifejezés inicializálja a ciklust, a második ellenőrzi a ciklus lejárát, míg a harmadik a ciklus egy-egy végrehajtása után módosítja a ciklusváltozók értékét. Például a BASIC-ből jól ismert $\text{FOR } i=1 \text{ TO } N \text{ STEP } 1$ ciklust a C-ben a következőképpen lehet kifejezni: $\text{for } (i=1; j++; j \leq n)$.

6. Egyéb utasítások. Ebbe a kategóriába összesen három utasítás tartozik. Ezek közül a legegyszerűbb az üres utasítás, ez a ;. Feleleges pontosvessző üres utasításként viselkedik. A

break utasítás befejezi az utasítást tartalmazó legbelső while, do, for vagy switch utasítást, s az azt követő első utasításra kerül a vezérlés. A continue utasítás befejezi a ciklusmag végrehajtását, s a ciklus folytatásának ellenőrzésére kerül vissza a vezérlés. Ez a kétféle do és a for ciklusra egyaránt vonatkozik.

Előre definiált függvények

Befejeztük a C nyelv ismertetésének első részét. A továbbiakban a típusdeklarációkat, az előre definiált függvényeket és a fordítási opciókat ismertetjük. Jelenlegi ismereteink azonban még nem elegendőek programok írására, ugyanis a C-ben az adat ki/beviteli műveletek mind előre megírt függvények (hasonlóan a PASCAL-hoz), ezért legalább néhány eljárást ismeretünk, hogy tudjunk már most programokat készíteni.

Adatbeolvasás

A billentyűzetről történő adatbeolvasás formája a következő:

```
scanf(<formátum-sztring>, {<argument> ...});
```

A formátum-sztring a beolvasandó értékek formátumát tartalmazza, míg az argumentum-lista a beolvasandó értékek helyét adja meg, vagyis mindegyik egy-egy mutató. A formátum-sztringben az egész értékeket a "%d" vagy a "%10d" formában kell jelezni. A % jel vezet be a vezérlő sorozatot, míg a "d" a decimális beolvasási formátumot jelzi. A köztük lévő szám a maximális beolvasandó jegyek számát jelenti. Bármilyen nem decimális jel véget vet a beolvasásnak. Például az x és y egész számokat a következőképpen lehet beolvasítani:

```
scanf("%d%d",&x,&y);
```

Kiírás a képernyőre

A képernyőre való kiírás ugyancsak formátum-sztring segítségével történik. Ennek alakja:

```
printf(<formátum-sztring>, {<argument> ...});
```

Argumentumokként most magukat az értékeket kell megadni, nem pedig a címüket. A <formátum-sztringben> található nem vezérlő karakterek egy az egyben megjelennek a képernyőn. A kiírás vezérlésére a következő karaktersorozatokat is lehet még használni:

C jelölés	hatás
\n	Új sor
\t	vízszintes tabulálás
\b	visszalépés
\r	kocsli-vissza
///	lapdobás (képernyő törés)
\\	backlash
\\	apozitról

Például a printf("\n\nEredmenyek: %d %d",x,y) hatására a képernyő törődik, s a második sor elején megjelenik az "Eredmenyek:" felirat, majd azt követően 5-5 helyiértéken ábrázolva az x és y számok értéke, köztük két szóközzel.

Most már kellő ismerettel rendelkezünk ahhoz, hogy elég bonyolult és összetett programokat írjunk C nyelven. Bemelegítőnek egy egyszerű példa.

Az alábbi program az első 1000 természetes szám között található prímszámot írja ki a képernyőre:

```
int oszthato(i,j)
{
  int l;
  for(l=2;l<=i;l++)
    if(i%l == 0) return -1; else return 0;
}
main()
{
  int i,j,flag;
```

```
printf("%1c%d\n",CLR,2);
for(i=2;i<1000;i+=2) {
  flag = 0;
  for(j=2;j<=i>1;(j+=2)?j++:(j+=2))
    if(flag = oszthato(i,j)) break;
  if(!flag)
    printf("%d\n",i);
}
}
```

Jelen ismertetőnkkel azzal fejezzük be, hogy ismertetjük, hogyan kell egy C programot megírni és futtatni: Töltsük be a C fordítót, majd bejelentkezés után nyomjuk meg az <EDIT> <ENTER> billentyűket. Ennek hatására szerkesztő üzemmódba kerülünk. Adjuk ki az H10,10 szerkesztő parancsot, majd ahogy a program sorban adja a számokat írjuk be az egyes programsorokat. Vigyázat: a C alapszavakat csak kis betűkkel írhatjuk be! Amikor valamennyit beírtuk, akkor a beszurási üzemmódból úgy tudunk kilépni, hogy egy olyan sort írunk be, amelyik egyedül az <EDIT> billentyű jelét (kis téglalap) tartalmazza. A fentiek elvégzése után nyomjuk meg még a <C> billentyűt is. Ennek hatására a szerkesztőből visszatérünk a fordítóba. A #INCLUDE <ENTER> után a fordító lefordítja a szerkesztő pufferében lévő programot. Ezután már csak a <SYMBOL SHIFT-I> (vagy AT) billentyűt kell megnyomnunk. Ha nem vettünk hibát, akkor a fordítótól azt az üzenetet kapjuk, hogy az <Y> gomb megnyomására elindul a program.

Elképzelhető, hogy mind a forrásnyelvi, mind a lefordított programot szalagra vagy mikrodrive-ra szeretnénk átmásolni. Ekkor a következőképpen kell eljárunk: ha a forrásnyelvi szöveget akarjuk eltávolítani, akkor a szerkesztőből ki kell adnunk egy P10,150, "ezanevem" parancsot. Vigyázzunk: a sorszámoknak a fordítóba a megírt program első és utolsó sorának a számával kell megegyeznie! Ha a lefordított programot akarjuk elmenteni, akkor a program szövegének első sorába a #translate <fájlnev> alakú vezérlő sort kell elhelyezni. A fájl mentésére a <SYMBOL SHIFT-I> billentyű megnyomásakor kerül sor. A lefordított és elmentett program visszatöltése után a RANDOMIZE USR 25200 BASIC parancsossal indítható el.

Rendben, nézzük, hogyan működik a program! Rögtön látszik, hogy két függvényből áll, ezek az oszthato és a main. A main a főprogram, ezért ennek nem lehet paramétere. Az oszthato nevű függvény kétváltozós és egész értéket ad vissza. Ez 0, ha i maradék nélkül osztható j-vel, különben -1. A main függvénynek három belső változója van, valamennyi egész. Ezek az i, j és a flag nevű változók.

A main kiírja az első prímszámot, ami a 2. (Ha valaki nem tudná: egy szám akkor prímszám, ha az 1-en és önmagán kívül maradék nélkül egyetlen pozitív számmal sem osztható...) Ezután következik egy ciklus, ami hárommal indul és 999-ig tart. A ciklusmag minden egyes lefutása végén végrehajtódik az i += 2 kifejezés kiértékelése, aminek következményeként i mindig kettővel nő. A belső ciklus ellenőrzi, hogy van-e az i-nek osztója. Ez a ciklus a j = 2 értékkel indul, s egészen i/2-ig tart. Az i/2 értéket trükkösen "i > 2"-nek írjuk be. A ciklusváltozót a (j = 2)? ++ : (j + 2) kifejezés segítségével módosítjuk. Ha j egyenlő kettővel, akkor j eggyel nő a ++ kifejezés hatására, minden más esetben kettővel, mert ilyenkor a j + 2 rész kerül kiértékelésre. A ciklusváltozó ilyen használata meggyorsítja a ciklust, hiszen a páros számok – kivéve a 2-t – kimeradnak. A ciklusmag egyetlen feltételes utasításból áll. A feltétel kiértékelésének egyik mellékhatása, hogy flag értéke mindig beállítódik. Ha a feltétel igaz (< > 0), azaz találtunk osztót, akkor a break miatt kilépünk a ciklusból. Végül ha a flag jelzi, hogy a számnak nincs osztója, akkor a második printf kiírja a számot. Ha még valaki itt van: írjunk olyan programot, amelyik kiírja egy szám prímtényezői felbonthatását. Használjunk minél bonyolultabb és trükkösebb értékadásokat!!!

LED STORM • GO!

Amikor a begyűjtött pontszámunk leszámolása történik, nyomjuk meg egyidejűleg a <CAPS SHIFT> és a <SPACE> billentyűket (BREAK), ekkor a képernyő kerete zöld színűre vált, a játék futása pedig felfüggesztődik. Most nyomjuk meg az aktuális tűz-gombot az újraindításhoz, lám 300.000 ponttal rendelkezünk.

TASK FORCE • CCS/Premier Software

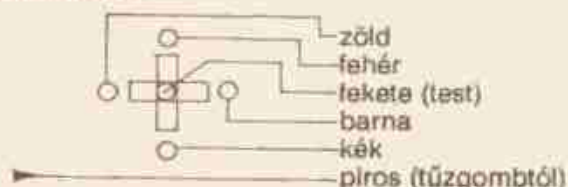
Amikor definiálnunk kell a billentyűzetet, gépeljük be sorban: C, H, E, A, T, majd ezt követően pedig definiáljuk a nekünk megfelelő billentyűket. Az induláskor egyetlen éllettel fogunk rendelkezni.

Numerikus billentyűzet

Köztudott, hogy a Spectrum billentyűzetén – még a 128+2, +3 billentyűzetén is – nehéz számokat, adatokat, a numerikus számítások során alkalmazott szimbólumokat (tizedespont, szorzásjel stb.) bevinni. Egy speciális célprogram (pl. bérelszámolás, bruttó-sítás stb.) esetén ez a probléma fokozottabban jelentkezhet. A 128K géptípusokhoz igaz forgalomba került egy numerikus külső billentyűzet, amely a KEYPAD csatlakozón keresztül illeszthető, ám úgy gondoljuk 48K-s gépből valamivel több található az országban, az ő problémájukat kellene elsősorban megoldani.

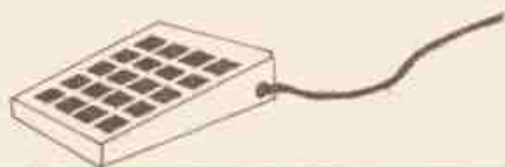
A megoldás kulcsa egy billentyűzetre programozható joystick interface – ilyenből sokféle létezik, még hazai forgalmazásban is –, egy lerobbant joystick, valamint a külső numerikus billentyűzet, melynek megépítésére most fogunk ötleteket adni.

Az első lépés a joystick szétszerelése. Célszerű, ha egy már egyébként használhatatlan joystick-et használunk fel erre a célra. A joystick kis nyáklapján meg fogjuk találni az oda csatlakoztatott kábelvégződést. Ez gyakran úgy néz ki, hogy minden ér más színű. Ezermester üzletekben kapható többeres szalagkábel (laposkábel), vegyünk ebből egy kb. 70 cm-es darabot. Ebből vágjunk le egy 20 cm-es részt, fejtünk le a belőle 6 különböző színű eret, majd pucoljuk meg a végeket, és forrasszuk az egyik oldali végződéseket a csatlakozásokhoz:



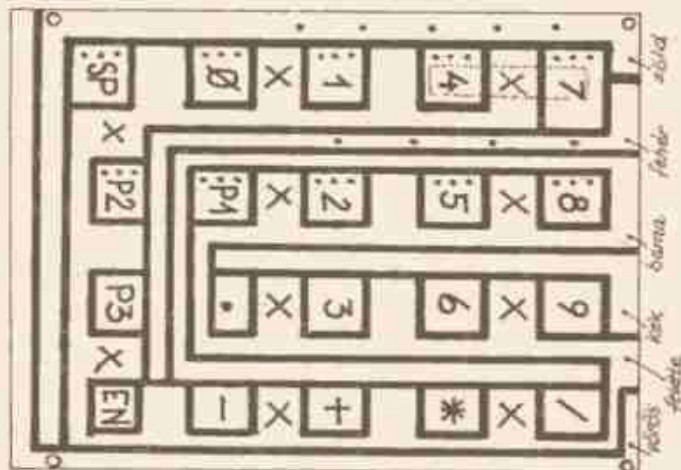
A szalagkábel másik végén is pucoljuk meg a vezetékvegeket, majd forrasszunk fel rá egy 6 pólusú tuchel dugót. A 6 pólusú tuchel dugót lengő aljzattal együtt szerezzük be, az aljzattal majd a numerikus billentyűzethez fogjuk felszerelni. Feltétlenül jegyezzük meg azt is, hogy melyik kábelszín melyik pólushoz tartozik.

Ezt követően szereznünk kell egy kisebb méretű műanyag dobozt. A doboz méretét persze a beépítés alapvetően befolyásolja, ám nem akarunk konkrét adatokat megadni (a x b x c), ez menet közben úgy is ki fog derülni. Lombfűrészsel vágjuk ki a billentyűk helyét, 10 x 10 mm-es nyílásokat, ugyanakkor vegyünk egy kb. 6-8 mm vastagságú gumilapot, ebből pedig vágjunk ki 8 x 8 mm méretű hasábokat (ezek lesznek a billentyűk).



A műanyag doboz felső lapjára (amelyen a nyílásokat kivágtuk) alulról vágjunk be egy megfelelő méretű gumilapot (célszerű rossze kerékpár belsőből), majd pillanatragasztóval a nyílásoknak megfelelően ragasszuk fel a billentyűket a vékony gumilapra. Amikor minden billentyű a helyére került, célszerű várni néhány órát pihenni, hogy a ragasztó kötési szilárdsága megfelelő legyen, ezután a számokat és a jeleket felvihetjük a billentyűk felületére. Ennek megoldását már Önökre bizzuk, akár égetéssel, akár festéssel.

Amíg a ragasztó megköt, nekiláthatunk a nyáklap elkészítésének. A kapcsolást egy – a vékony gumilappal megegyező nagyságú nyáklapon fogjuk megtervezni, ez házilag is könnyen megoldható. A nyák felületén a vastag sávokat maratjuk ki, a szín-jelölések az általunk elkészített billentyűzetre vonatkoznak, lehet bármilyen más vezeték-szín kombináció.



A számozott négyzetek szélén és a csikokon készítünk \emptyset 1 mm-es furatokat, a szerkezet szívére képező egyenirányító diódák részére, ugyanis ezekkel lehet megoldani, hogy a joystick-kel ellentétben, a joystick átlós irányának megfelelő kettős kapcsolást megkapjuk. Ide már a leggyengébb egyenirányítók is megfelelnek, kb. 7,- Ft/db, és 35 db-ra lesz szükségünk. A nyákhöz forrasszuk a szalagkábel másik részét (kb. fél méter), a színeknek megfelelően, a másik végére pedig szereljük fel a tuchel lengő aljzattal. A diódákat a kis furatokon keresztül a nyák síma oldala felől felidugva forrasszuk be. A fő irányoknak (2,4,6,8) megfelelő kapcsolásokat dióda nélkül, azokból lecsipett darabokkal oldhatjuk meg. A fő irányok betartása fontos a joystick betanításához! A diódákat úgy forrasszuk fel, hogy az azokon látható "csik" jelzés mindig a kiinduló mezőhöz (pl. 7,9) essen közelebb. Az irányába mutasson. Részletesebb rajz helyett itt leírjuk, hogy mely mezőtől mely színhez csatlakozzon a dióda.

8	– fehér	dióda nélkül
4	– zöld	dióda nélkül
2	– kék	dióda nélkül
6	– barna	dióda nélkül
5	– piros	dióda nélkül
7	– fehér	diódával

7	- zöld	diódával
9	- fehér	diódával
9	- barna	diódával
1	- zöld	diódával
1	- kék	diódával
3	- kék	diódával
3	- barna	diódával
0	- zöld	diódával
0	- fehér	diódával
0	- piros	diódával
/	- kék	diódával
/	- barna	diódával
/	- piros	diódával
*	- fehér	diódával
*	- barna	diódával
*	- piros	diódával
+	- zöld	diódával
+	- kék	diódával
+	- piros	diódával
-	- zöld	diódával
-	- piros	diódával
SP	- fehér	diódával
SP	- piros	diódával
EN	- barna	diódával
EN	- piros	diódával
P1	- kék	diódával
P1	- piros	diódával
P2	- fehér	diódával
P2	- kék	diódával
P3	- zöld	diódával
P3	- barna	diódával
.	- zöld	diódával
.	- barna	diódával
.	- piros	diódával

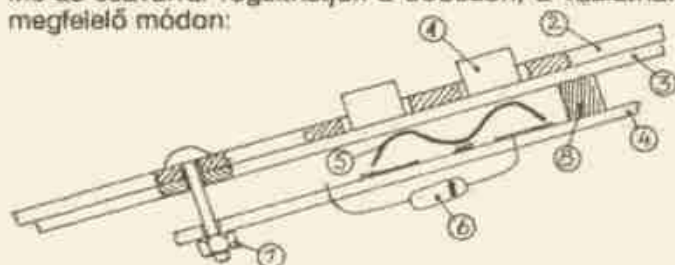
Ahol SP = SPACE, EN = ENTER, ill. P1, P2 és P3 tetszőleges billentyűk.

Amikor ez is kész lett, vágjunk le rugalmas rézlemezről kb. 2 mm-es csíkokat, egyenként kb. 20 mm hosszút, majd hajlítsuk meg ezeket az ábrának megfelelő alakra.



Forrasszuk ezeket a csíkokat a nyákrájon X-szel jelölt, tehát a közös (fekete) érintkezőkre. Attól függően, hogy a hajítás milyen magasra sikerült, ragaszunk a gumilap aljára hálószerűen megfelelő magasságú gumicsíkokat, ez fogja megadni a billentyűzet érzékenységét, és meggátolja, hogy másik kapcsoló is bekapcsoljon.

Most már a billentyűket belülről feinyomhatjuk a műanyag dobozon kivágott nyílásokba, a nyákat 4 db. M3-as csavarral rögzíthetjük a dobozon, a vázlatnak megfelelő módon:



- | | | | |
|---|----------------------|---|---------------------|
| 1 | - billentyű | 5 | - rugalmas lemez |
| 2 | - műanyag doboz | 6 | - dióda |
| 3 | - gumilap | 7 | - csavar |
| 4 | - nyomtatott áramkör | 8 | - távtartó gumicsík |

Amennyiben az összeszerelés rendben megtörtént, következhet a főpróba! A tanítható, programozható joystick interface-t tanítsuk be úgy, mintha a joystickkal tennénk, ám most a numerikus billentyűzet billentyűi segítségével. A számok, műveleti jelek, a tizedespont, a SPACE és az ENTER mellett rendelkezésre álló 3 billentyűt a célprogram sajátosságai szerint választhatjuk ki, tetszés szerint.

Reméljük, hogy hasznos információval láttuk el mindazokat, akik egy kicsit is szeretnek otthonukban barakácsolni, és gépüket gyakran használják numerikus feladatokra. A felhasználáshoz sok sikert kívánunk!

KILLED UNTIL DEAD

LEVEL IV. SUPERSLEUTH

1. pálya

(Last Laff)

gyilkos: Claudia

áldozat: Sydney

hely: Agatha's room

egyver: bomb

ok: Sidney said you were

Cousy writer

Break In

Sydney - 4

Peter - 2

Calaudia - -

Agatha - Phillis Dorothy

Mike - -

2. pálya

(Motherly Love)

gyilkos: Agatha

áldozat: Sydney

hely: patio

egyver: knife

ok: You wanted Mike's

inheritance

3. pálya

(Rhymes and crimes)

gyilkos: Claudia

áldozat: Agatha

hely: library

egyver: gun

ok: Jealousy

4. pálya

(The Scales of Justice)

gyilkos: Claudia

áldozat: Peter

hely: patio

egyver: poison

ok: He weighs less than
the rest

Break In

Sydney - Beekeeper

Agatha - Rammsers

Claudia - Danger Man

Mike - E.A.Poe

Peter - -

Uége!

A billentyűzet figyelése

A SpV. 15. számában a billentyűzet figyelésének BASIC-ből megvalósítható módszerére szemléltettünk egy példát. Sajnos elkövettük egy hibát: a *Beta Basic* segédprogrammal nem mindenki rendelkezik. Sok levélíró is javasolta: ne tegyük feltétlenül szükségessé a *Beta Basic* betöltését, ha a módszer az alap BASIC utasításkészletével is szemléltethető. Ezért most a rutint egy kicsit átírtuk.

```

10 REM *** Bill.figyeltes ***
20 DIM b$(5): FOR i=16 TO 175
STEP 32: PLOT 0,1: DRAW 255,0: N
EXT i
25 PLOT 0,16: DRAW 0,153: DRAW
255,0: DRAW 0,-153
30 PRINT AT 2,5;"A billentyuk
figyelese"
40 PRINT AT 5,1;"1-5":AT 6,1;6
3486;AT 9,1;"Q-T":AT 10,1;64510;
AT 13,1;"A-G":AT 14,1;65022;AT 1
7,1;"CS-V":AT 18,1;65278
50 PRINT AT 5,28;"0-6":AT 6,26
;61438;AT 9,28;"P-Y":AT 10,26;57
342;AT 13,26;"ENT-H":AT 14,26;49
150;AT 17,27;"SP-B":AT 18,26;327
66
60 PLOT 128,144: DRAW 0,-128:
PLOT 50,144: DRAW 0,-128: PLOT 3
03,144: DRAW 0,-128
70 LET a=IN 61438: PRINT AT 5,
18;a: GO SUB 100: PRINT AT 6,8;b
$: LET a=IN 64510: PRINT AT 9,10
;a: GO SUB 100: PRINT AT 10,8;b$
: LET a=IN 65022: PRINT AT 13,10
;a: GO SUB 100: PRINT AT 14,8;b$
: LET a=65278: PRINT AT 17,10;a:
GO SUB 100: PRINT AT 18,8;b$
80 LET a=IN 61438: PRINT AT 5,
18;a: GO SUB 100: PRINT AT 6,18;
b$: LET a=IN 57342: PRINT AT 9,1
8;a: GO SUB 100: PRINT AT 10,18;
b$: LET a=IN 49150: PRINT AT 13,
18;a: GO SUB 100: PRINT AT 14,18
;b$: LET a=IN 32766: PRINT AT 17
,18;a: GO SUB 100: PRINT AT 18,1
8;b$
90 PRINT AT 20,0;"Aktualis bill
entyu: ";CHR$(PEEK 23560);"
*
95 POKE 23560,32
100 LET b$="000000"
110 LET x=a-160
120 IF x/16<1 THEN LET b$(5)="1"
*
130 IF x/16>=1 THEN LET x=x-16
140 IF x/8<1 THEN LET b$(4)="1"
150 IF x/8>=1 THEN LET x=x-8
160 IF x/4<1 THEN LET b$(3)="1"
170 IF x/4>=1 THEN LET x=x-4

```

```

180 IF x/2<1 THEN LET b$(2)="1"
190 IF x/2>=1 THEN LET x=x-2
200 IF x<1 THEN LET b$(1)="1"
210 RETURN

```

Ellipszis kompozíció

Ez az egyszerű demonstrációs program azt mutatja be, hogy különböző koordinátájú ellipszisek segítségével hogyan tudunk 3 dimenziós hatást kelteni:

```

1 CLEAR
2 INK 2
5 LET x=0: LET y=80
10 FOR n=0 TO 2*PI STEP PI/180
15 PLOT 128+x*SIN n,87+y*COS n
20 NEXT n
25 LET x=x+10: LET y=y-10
30 IF y=-10 THEN STOP
40 GO TO 10

```

Kristály kompozíció

A trigonometrikus függvények és a SPECTRUM rajzoló utasításainak együttes felhasználásával szép, gyémántcsiszolathoz hasonló ábrát kapunk eredményül:

```

10 CLEAR
20 DIM a(12): DIM b(12)
30 FOR n=1 TO 12
40 LET k=n/6*PI
50 LET a(n)=128+80*SIN k: LET
b(n)=88+80*COS k
60 PLOT a(n),b(n)
70 NEXT n
80 FOR n=1 TO 12
90 FOR m=1 TO 12
100 LET ox=a(m)-a(n)
110 LET oy=b(m)-b(n)
120 PLOT a(n),b(n): DRAW ox,oy
130 NEXT m: NEXT n

```

Plusz egy grafika

Ez egy igazi ABS-trakt alkotás, ugyanis az ABS függvény alkalmazásával értük el ezt a meghökkentően érdekes hatást!

```

10 REM ABS-trakt
20 FOR k=0 TO 60 STEP 10
30 FOR x=0 TO 128
40 LET y=30-ABS(x-32)*SIN(x*
PI/64)
50 PLOT x,y+k: PLOT y+k,x
60 PLOT 255-x,y+k: PLOT 255-(y
+k),x
70 NEXT x
80 NEXT k

```

SHANGHAI WARRIORS • Players

Amikor feliratkozunk a HIGH SCORE táblára, a nevünk helyett gépeljük be: OUTLAND. Ha most elindítunk egy új játékot, rendelkezésünkre fog állni egy bomba, amelyet minden időben aktivizálhatunk.

SKATEBALL • Electronic Arts

A címképernyőn gépeljük be: TIXY, akkor a gép készségesen végigmutatja nekünk mind a 26 szintet majd végtelen élettel ajándékoz meg bennünket.

Az amatőr programozók döntő többsége csupán a BASIC nyelvet ismeri és használja, holott szívesen beillesztené egy-két gépi kódú rutint is programjaiba olyan feladatok elvégzésére, amelyek BASIC nyelven csak igen dögösen, túlságosan lassan, vagy egyáltalán nem oldhatók meg. Többek között ehhez kívántak segítséget nyújtani a Gépi kód tanfolyam sorozat előző fejezetei, amelyek a SPECTRUM-ban is alkalmazott Z-80 mikroprocesszor utasításkészletét ismertették több-kevesebb részletességgel. Ezek az utasítások képezik a gépi kódú programozás alapélemeit – mondhatnánk építőköveit – amelyekből a program felépíthető. A BASIC utasításokhoz szokott programozó azonban ezeket nem tudja a megszokott programozási gondolatmenetbe beilleszteni, így – hacsak nem kap segítséget – szinte nehezebben igazodik el közöttük, mint a BASIC nyelvet sem ismerő kezdő.

A segítség legjobb és legegyszerűbb módja néhány mintaprogram bemutatása, az alkalmazott gondolatmenet levezetésével. A következőkben ismertetésre kerülő programok ill. rutinok főleg ezt a célt szolgálják, de olyan gyakorlati feladatok megoldására irányulnak, amelyek amatőr BASIC programokban is sokszor előfordulnak, a kidolgozás viszont olyan, hogy ezek a rutinok meglévő amatőr programokba is beilleszthetők.

Bevetetőül mindenesetre néhány megfontolásra érdemes jó tanács:

1. Tanuljuk meg valamelyik ASSEMBLER/DISASSEMBLER program kezelését, és lehetőleg rendszeresen azt használjuk. Nálunk legelterjedtebb a GENS-3, MONS-3 páros, amelyek nagy előnye, hogy a memória tetszőleges területére tölthetők be (a mintaprogramok is erre készültek, de ez nem zárja ki más monitor-assembler program használatát). Ha mindkettőt betöltjük, akkor az assembleri program azonnal ellenőrizhető és futtatható.
2. Legyen kéznél a Z-80 utasításkészlet összefoglaló jegyzéke (mit fogad el a mikroprocesszor?) és részletes ismertetője, amelyeket bizony eleinte szinte minden utasítás beírása előtt segítségül kell hívnunk.
3. Az assembler programba beírt utasítássorozatot (forráskódot) célszerű futtatás előtt kimenteni, amit minden assembler program lehetővé tesz.
4. Ne feledkezzünk meg a RAMTOP áthelyezéséről CLEAR utasítással, a gépi kódú programunk alá, különösen ha a memória legfelső részét is használni akarjuk. A ROM program ugyanis közvetlenül a RAMTOP alatti rekeszeket használja veremterületként, aminek felülírása kellemeden következményekkel jár.
5. Folyamatábra készítése nemcsak ajánlatos, de összetett program esetén az áttekinthetőség csak így biztosítható.

A bemutatásra kerülő mintaprogramok főként demonstrációs célt szolgálnak, a legegyszerűbb utasításokból összeállítva. Minden más szempont – pl. tömörség, működési sebesség, kis helyszükséglet – háttérbe került.

Kezdjük a gyakorlást egy olyan rutinnal, amely egy rendezetlen számsor tagjait nagyság szerinti sorrendbe rendezi. Egyszerűség kedvéért a számsor legfeljebb 255 tagból álljon, és az egyes számértékek is csak 0..255 közötti pozitív egész számok lehetnek (így ul. csak egy byte hosszúságú adatokat kell kezelni).

A rendezés gondolatmenete a következő:

A számsor első tagját egymás után összehasonlítjuk az összes mögötte állóval, és ha valahol egy nála kisebb értéket találunk, a két számot felcseréljük. Csere után folytatjuk a műveletet, de már ahhoz az új értékhez hasonlítunk, amit csere után az első helyre beírtunk. A sor végére érve bizonyos, hogy a számsor elején annak legkisebb értékű tagja áll. Ezt megismételve a második, harmadik stb. taggal, a rendezés teljesse válik. A programot a memóriában bárhol elhelyezhetjük. Ha nem használunk nyomtatót, akkor kényelmes megoldás a nyomtató 23296. címen kezdődő, 256 byte hosszú tárterületének felhasználása, ahol a programot semmi nem zavarja, és még a RAMTOP-ot sem kell áthelyezni.

Számrendezés

10	TAGOK	EQU	23296
20	CIM	EQU	23298
30	CIMTAR	EQU	23300
40	CIKLUS	EQU	23302
50		ORG	23310
60		LD	HL,(CIM)
70		LD	A,(TAGOK)
80		DEC	A
90		LD	B,A
100	C1	LD	A,B
110		LD	(CIKLUS),A
120		LD	A,(HL)
130		PUSH	HL
140	C2	INC	HL
150		CP	(HL)
160		CALL	NC,CSERE
170		DJNZ	C2
180		POP	HL
190		INC	HL
200		LD	A,(CIKLUS)
210		LD	B,A
220		DJNZ	C1
230		RET	
240	CSERE	LD	C,(HL)
250		LD	(HL),A
260		LD	(CIMTAR),HL
270		POP	DE
280		POP	HL
290		PUSH	HL
300		PUSH	DE
310		LD	(HL),C
320		LD	HL,(CIMTAR)
330		LD	A,C
340		RET	

Az egyes lépések magyarázata:

A 10-50 sorok csupán ún. assembler direktívák, amelyek a programozást könnyítik. Egyszerűbb ugyanis címkéket beírni mint számértékeket, sőt esélyünk van arra, hogy tévedés esetén az assembler program erre figyelmeztet. Ezekben:

- 10 - a számsor tagjainak számát tároló rekesz címe;
- 20 - a számsor kezdőcímét tartalmazó rekeszpár;
Az előbbi adatokat nem rögzítjük a programban, hanem esetenként kívülről adjuk meg (BASIC-ből POKE utasítással), hogy a programunk univerzálisan használható legyen;
- 30 - itt a mindenkor soron következő szám címét helyezzük el (2 byte);
- 40 - ez a ciklusváltozó (még hátralévő tagok száma) tárolási helye;
- 50 - A program kezdőcíme;
- 60 - Lehívjuk a számsor kezdőcímét;
- 70 - és a tagok számát (n);
- 80 - de csak n-1 lépést kell tenrünk
- 90 - Ezt azért töltjük át B-be, mert a most következő ciklusban a B értéket használjuk, ciklusváltozóként.
- 100 - a C1 külső ciklus kezdete, a B-ben hordozott ciklusváltozót minden visszatéréskor a memória adott rekeszében kívánjuk elraktározni, de oda csak az A regiszterből tudunk értéket betölteni (110).
- 120 - Mivel itt HL mindig arra a címre mutat (és ügyelünk arra, hogy ez a ciklusból való visszatérés után is így legyen), ahol a soron következő báziszám található, az ott lévő értéket az A regiszterbe töltjük;
- 130 - a címet pedig kimenjük a verembe. (A verembe való adattárolás mindig nagyobb körültekintést igényel, mint a memóriarekeszben való tárolás, mivel onnan az adatok csakis egymás után, fordított sorrendben vehetők ki).
- 140 - A belső, C2 ciklus kezdete, amelyben a báziszámot rendre összehasonlítjuk a mögötte állókkal. Először a következő tag címére lépünk (140), és az ott lévő értéket összehasonlítjuk az A-ban lévő báziszámmal (150).

- 160 - Ha ez az érték a báziszámnál nem nagyobb, vagyis nem volt átvitel (amit a C jelzőbit mutat), akkor meghívjuk a CSERE szubrutint. (Vegyük észre, hogy a program azonos értékek esetén is cserél, de ez az eredményt nem befolyásolja).
- 170 - B értéke csökken, és amíg 0 értéket el nem éri, vissza a ciklus kezdetére. Fontos tudni, hogy a B regisztert közben nem használtuk, tehát abban még a ciklusváltozó értéke van.
- 180 - A külső ciklust a következő számmal folytatjuk, amelynek címét a veremből kivett érték növelésével kapjuk (egy lépés előre).
- 200 - Visszatérés előtt leolvassuk és B-be töltjük a ciklusszámot (210), és értéket csökkentve visszatérünk a ciklus elejére, ha még van hátralevő tagunk (220). Egyébként vége (230). A CSERE szubrutinba lépéskor a HL regiszterpar az éppen vizsgált szám címét tartalmazza, míg a báziszám címe a címtárban, értéke viszont az A regiszterben van. Ezeket kell egymással felcserélni.
- 240 - Átmenetileg C-be töltjük a vizsgált szám értékét;
- 250 - helyére beírjuk a báziszámot;
- 260 - és megőrizzük azt a címet is, ahol a csere véget értünk. Az aktuális kezdőcímet, ahová C értéket be kell töltenünk, a veremben tároljuk, tehát onnét kell kivenni. Igen ám, de most egy szubrutinban vagyunk, és az ide lépéskor a ROM program a verem tetején helyezte el a visszatérési címet, alatta helyezkednek el az oda korábban bevitt értékek. Ezért:
- 270 - Kivesszük a felül lévő adatot bármely használaton kívüli regiszterpárba,
- 280 - majd utána a keresett címet is, és visszaállítjuk az eredeti állapotot (290, 300), perazze ügyelve a fordított sorrendre.
- 310 - Az aktuális kezdőcíme beírjuk az új értéket.
- 320 - és a programot ott folytatjuk, ahol megszakítottuk.
- 330 - de már az új báziszámmal.

Következő programunk célkitűzése legyen egy rendezett számsor véletlenszerű, alapos összekeverése. Ez az igény nem csak az amatőr játékprogramokban gyakorol, hanem a számítástechnika egyéb ágában is. Mi azért maradunk meg a kezdő amatőr szintjén, és ennek megfelelően választjuk ki a követendő eljárást is.

Ismét vezessük be azt a korlátozást, hogy a számsor csak egy byte-os egységekből áll, és tagjainak száma sem több 255-nél. Elhelyezünk egy n tagú számsort a memória egymást követő rekeszeiben, és egy alkalmas üres területet feltöltünk egy ugyancsak n számú, INT RND n tagokból álló véletlen számsorral, amit BASIC-ből könnyen és gyorsan lehet generálni. (A helypazarlást tekintjük bocsánatos bűnnek). Ezek után a keverés annyiból áll, hogy végiglépve a számsoron, valamint az RND számsoron is, az utóbbi helyén álló véletlenszám azt adja meg, hogy a számsor adott tagját melyikkel (a sor elejétől számítva hanyaddikkal) kell felcserélni. Így elvileg minden tagot megmozgatunk, és jó átkeverés az eredmény.

A fő paramétereket ezúttal is kívülről visszük be, tehát a program univerzálisan használható.

Keverés

10	CIM	EQU	23296
20	RND	EQU	CIM+2
30	NN	EQU	CIM+4
40	TAG	EQU	CIM+6
50		ORG	23310
60		LD	BC, (CIM)
70		LD	DE, (RND)
80		PUSH	BC
90		LD	A, (NN)
100		LD	B, A
110	CIKL	LD	A, B
120		LD	(TAG), A
130		POP	BC
140		LD	A, (BC)

150	PUSH	AF
160	PUSH	BC
170	LD	A, (DE)
180	LD	C, A
190	LD	B, 0
200	LD	HL, (CIM)
210	ADD	HL, BC
220	LD	A, (HL)
230	POP	BC
240	LD	(BC), A
250	POP	AF
260	LD	(HL), A
270	INC	DE
280	INC	BC
290	PUSH	BC
300	LD	A, (TAG)
310	LD	B, A
320	DJNZ	CIKL
330	RET	

A program működése:

Az assembler direktívákkal kezdjük ismét a beírást, ahol

- 10 - a számsor kezdőcíme,
- 20 - a véletlen sor kezdőcíme,
- 30 - a sor tagjainak száma és
- 40 - a ciklusváltozó tárolási helye.

A számsor kezdőcímét a BC-be, az RND sor címét DE-be töltjük, majd a kezdőcím verembe való kimentése után, a tagok számát írjuk be a már ismert módon a tárolás helyére (120).

Az aktuális címet a veremből vesszük ki (130), és a cserélendő szám ott lévő értékét félreteesszük a verembe (150), ahová a soros címet is visszatesszük (180).

A DE címen lévő véletlenszámot a kezdőcímehez kell hozzáadni, hogy ezt könnyen megteheszük, az értéket egy megfelelő regiszterpárba kell tölteni, ami csak több lépésben lehetséges, majd az összeadást elvégezve (210) HL-ben már a csereszám címe van. Az innen kivett számértéket (220) a veremből kivett aktuális címre betöltjük (240), helyébe pedig azt az értéket tesszük, amit előzőleg az aktuális címről a verembe mentettük (280).

A számcseré végrehajtása után mindkét soron egyet lépünk előre, a BC-ben lévő soron következő címet kimentjük a verembe, mert a BC regiszterpárt közben mára is használjuk. A ciklusváltozó lehívása (300) után, értéket csökkentjük, és amíg nem éri el a 0-t, tovább folytatjuk a műveletet.

A programozást kedvelő amatőrök közül kevesen tudnak ellenállni annak a csábításnak, hogy elkészítsék a közismert számítástáblás logikai feladvány programját, melynek az a lényege, hogy egy 4x4 mezőből álló táblán lévő 15 számot rendezetlen állapotból, tologatással, rendezett állapotba kell hozni. (Számítógépen megoldható a más méretű, pl. 3x3 vagy 5x5 mezőből álló tábla előállítás is, ami a játékot változatosabbá teszi).

A valamikor közkezdvelt mechanikai felépítésű táblán (eredeti neve a feltaláló után "Lloyd") minden állásból rendbe lehetett hozni a számokat, hiszen az állás az eredetileg rendezett állapotból, ugyancsak tologatással jött létre, ha azonban a rendezetlen állapotot véletlenszerű keveréssel állítjuk elő, akkor matematikailag törvényszerűség, hogy a megoldhatóság valószínűsége 50%. Az amatőr programok nagy része – még az egyébként meglepően színvonalasak is – hibás, ugyanis ezt a körülményt nem veszi figyelembe. A megoldás többek között a Spencer: "Játékok BASIC nyelven" címmel magyarul is kiadott könyvben megtalálható. Az ellenőrző eljárás több műveletből áll:

1. A kevert számsor első tagjától kiindulva (majd ugyanezt minden egyes tagra megismételve) megszámoljuk, hogy hány nála kisebb értékű tag áll mögötte, és ezek számát összegezzük (az üres mezőt eredeti értékével, tehát pl. 4*4-es tábla esetén 16-tal kell figyelembe venni).
2. A táblát a sakktáblához hasonlóan osszuk be sötét és világos színű mezőkre. Ha az üres mező színe nem egyezik meg a jobb alsó sor színével, akkor az előbb kapott értékhez még adjunk hozzá 1-et.

3. Ha az így kapott érték páros, akkor a tábla rendezhető. Egyébként két szomszédos számot (de nem az üres mezőt!) kell egymással felcserélni, és a számsor rendezhetővé válik.

Mintaprogramunk az 1. műveletre vonatkozik, aminek BASIC változata meglehetősen bonyolult és lassú, az eljárás 2. és 3. pontjait a játéprogramba kell beépíteni.

Kontroll

10	CIM	EQU	23296
20	NN	EQU	CIM+2
30	SORCIM	EQU	CIM+4
40	SUMMA	EQU	CIM+6
50	ORG		23310
60	LD	IX,SUMMA	
70	LD	(IX+0),0	
80	LD	HL,(CIM)	
90	LD	A,(NN)	
100	DEC	A	
110	LD	B,A	
120	C1	PUSH	BC
130	LD	(SORCIM),HL	
140	LD	DE,(SORCIM)	
150	C2	INC	DE
160	LD	A,(DE)	
170	CP	(HL)	
180	JR	NC,UGR	
190	INC	(IX+0)	
200	UGR	DJNZ	C2
210	INC	HL	
220	POP	BC	
230	DJNZ	C1	
240		RET	

A program működése:

Az összegző memóriarekesz nullázása (70) után HL-be a számsor kezdőcímét, A-ba a tagok számát töltjük. Ez utóbbinál 1-gyel kevesebb a szükséges lépések száma (100), és ez lesz a ciklusváltozó, amit a B regiszter hordoz (110).

A C1 ciklus első lépéseként a B-ben lévő ciklusváltozót mentjük ki a verembe, majd a HL-ben lévő soron következő címet töltjük a tárolási helyére (130). Innen vesszük ki a mindenkor soros cím értékét (140).

A C2 ciklus minden fordulóban egyet előre lépve (150) az ott talált értéket hasonlítja össze a soros, HL-ben lévő címen található bázisszámmal (170), és ha ez nagyobb értékű, akkor a C jelzőbit értéke 1 lesz, és a 180 sorból nincs ugrás, tehát a számláló rekesz tartalma 1-gyel növekszik (190). A ciklus a hátralévő tagok számától függően ismétlődik, majd visszatérünk a C1 ciklusba, és a következő tagra végezzük el a vizsgálatot.

A végeredményt a SUMMA nevű rekeszben találjuk, és az már a játékprogram dolga, hogy ezt hogyan használja fel.

Aki figyelmesen követte az eddigi mintaprogramok gondolatmenetét, az bátran vállalkozhat egy összetettebb feladat megoldására is, méghozzá az eddigieknél szüksézebb magyarázatokra támaszkodva.

Legyen az új feladat egy olyan program, amely szöveges adatok (névsor, katalógus, címtár stb.) ABC sorrendbe való rendezésére alkalmas. Ennél a témánál különösen szembetűnő a BASIC és a gépi kódú programok sebessége közötti különbség. A stringek legyenek egyforma hosszúak, és helyezkedjenek el a memória adott címétől kezdve egymás után, egymástól azonos távolságra. A gondolatmenet az egyszerű számszámrendező rutinhoz hasonló: Az első string első karakterét összehasonlítjuk a mögötte állók első karakterével, és attól függően cserélünk vagy továbblépünk, hogy a két kódszám közül melyik nagyobb. Egyenlőség esetén az adott stringek első, második, harmadik stb. karaktereinek összehasonlítását kell elvégezni.

A program készítésénél vegyük figyelembe, hogy az ilyen jellegű listák sorai (többnyire 32 karakter koronként) általában sorszám-
bó), címből és az ezt követő jellemző adatokból állnak. Rende-

zőskor a sorszám a helyén marad, és csak a címet kell sorba rendezni, de csere esetén a címmel együtt mozgatjuk a jellemző adatokat is.

Szörendező

10	NN	EQU	65001
20	KEZD	EQU	NN+2
30	NN	EQU	NN+4
40	CIM	EQU	NN+6
50	SZAM	EQU	NN+8
60	TAG	EQU	NN+10
70	KCIM	EQU	NN+12
80	SCIM	EQU	NN+14
90	STAG	EQU	NN+16
100	BETU	EQU	NN+18
110	KAR	EQU	NN+20
120	ORG		65100
130			EXX
140			PUSH HL
150			EXX
160	LD	HL,(KEZD)	
170	LD	DE,(SZAM)	
180	ADD	HL,DE	
190	LD	(KCIM),HL	
200	LD	BC,(NN)	
210	DEC	BC	
220	C1	LD	(TAG),BC
230	LD	HL,(KCIM)	
240	LD	(SCIM),HL	
250	C2	LD	(STAG),BC
260	LD	HL,(KCIM)	
270	LD	A,(HL)	
280	LD	HL,(SCIM)	
290	LD	DE,(HM)	
300	ADD	HL,DE	
310	LD	(SCIM),HL	
320	CP	(HL)	
330	JR	NZ,U1	
340	CALL	BELSŐ	
350	U1	CP	(HL)

; ismételt összehasonlítás

360 JR C,U2
: és ugrás, ha (HL) > A
370 CALL CSERE
: Egyébként szöcsere meghívása
380 U2 LD BC, (STAG)
: BC-ben a C2 ciklusváltozója
390 DEC BC
400 LD A,B
410 OR C
: Megjegyzés a program végén!
420 JR NZ,C2
: Vissza az elejére, amíg BC nem egyenlő
: (0)-val. Ha nincs tovább, a C1 ciklus
: folytatható.
430 LD HL, (KCIM)
: Aktuális kezdőcím
440 LD DE, (MM)
: és lépésköz lehívása
450 ADD HL, DE
: Így kapjuk a következő kezdőcímet,
460 LD (KCIM), HL
: amit a tároló rekeszben megőrzünk.
470 LD BC, (TAG)
: C1 ciklusváltozó lehívása
480 DEC BC
490 LD A,B
500 OR C
510 JR NZ,C1
: Vissza amíg BC nem egyenlő (0)-val,
: egyébként a program befejezhető.
520 EXX
530 POP HL
540 EXX
550 RET
560 BELSO LD DE, (KCIM)
: További karakterek
: összehasonlításához
570 LD HL, (SCIM)
: az indulási adatok
580 LD A, (CIM)
: lehívása.
590 DEC A
: Csak A-1 lépést kell tenni.
600 LD B,A
610 C3 LD A,B
: Ez a ciklusváltozó,
620 LD (BETU), A
: amit megőrzünk
630 INC DE
: Következő karakter helye
640 INC HL
: mindkét címben
650 LD A, (DE)
: és az ott talált értékek
660 CP (HL)
: összehasonlítása
670 JR NZ,U3
: Ugrás, ha A nem egyenlő (HL)-tel

680 LD A, (BETU)
: Lehívjuk a C3 ciklusváltozót
690 LD B,A
700 DJNZ C3
: B-1, és vissza ha B nem egyenlő 0-val,
710 RET
: egyébként vége.
720 CSERE LD DE, (KCIM)
: Szöcsere szubrutin induló adatainak
730 LD HL, (SCIM)
: betöltése.
740 LD A, (SZAM)
750 LD B,A
760 LD A, (MM)
: A-ban egy teljes sor hossza
770 SUB B
: amiből a sorszám hosszát kivonjuk
780 LD B,A
: és ebből lesz B-ben a
790 C4 LD A,B
: C4 ciklusváltozó,
800 LD (KAR), A
: amit megőrzünk
810 LD A, (DE)
: DE-ben még a KCIM tartalma van,
820 LD B,A
830 LD A, (HL)
: HL-ben pedig az SCIM-é
840 LD (DE), A
: és a két címen levő értékeket
850 LD (HL), B
: egymással felcseréljük.
860 INC HL
: Következő karakter címe
870 INC DE
: mindkét címben.
880 LD A, (KAR)
: C4 ciklusváltozó
890 LD B,A
900 DJNZ C4
: B-1, és vissza, amíg B nem egyenlő 0-val
910 RET
: egyébként a szubrutin vége.

Megjegyzés: a 390...420, továbbá a 470...510 utasítások során lényegében a BC regiszterpárban hordozott ciklusváltozó értékét csökkentjük, majd ellenőrizzük, hogy ez elérte-e már a 0 értéket (tehát B=0 és C=0). Az itt alkalmazott frappáns megoldás az OR logikai utasításának azt a sajátosságát használja ki, hogy annak elvégzése után (410) az A regiszter értéke csak akkor lesz 0, ha előzőleg mindkét változó – vagyis az A-ba töltött B, valamint a C értéke 0 volt.

A program, természetesen, bárhol elhelyezhető, akár az eddigiekhez hasonlóan a nyomtató puffterületén, de ne feledjük, hogy az ilyen jellegű programokhoz gyakran használnak nyomtatót. A 10..50 adatokat kívülről kell bevinni, ami az univerzális alkalmazást teszi lehetővé.

Painter

A POKE-olást a BASIC/22800/25 file térképpel rendelkező verzióra fogjuk közölni

MERGE ~-dzzeljük be a BASIC betöltőt, majd állítsuk meg a magnót, és várjunk kb. 9 másodpercet, türelmesen. Ekkor megjelenik az OK. üzenet. A 2. sorból töröljük ki a 2 db. POKE utasítást, majd EDIT-eljük a 620-as sort LIST 620 (ENTER), CAPS SHIFT - 1

A 620-as sorban a „liv” érték adja meg az életék számát, ide írhatunk bármennyit, pl. 500, vagy 1-et is.

Adjuk ki: RUN (ENTER), majd töltsük be a 2. kódot, és annyi életünk lesz, amennyit a 620-as sorba beírtunk.

Gyakorlatilag az örökéletet is hasonló módszerrel érhetjük el, annyi különbséggel, hogy a 620-as sor EDIT-álása helyett a 2020-as sort kell kitérőlnünk.

Tartalomjegyzék

1	<i>Bevezetés helyett</i>	1
2.	<i>Játék, POKE, térkép</i>	2
	- <i>SOLDIER OF FORTUNE, NETHERWORLD</i>	2
	- <i>FERNANDEZ MUST DIE...</i>	3
	- <i>...SIR FRED</i>	4
	- <i>R-TYPE</i>	5
2.1	<i>Heavy On the Magick! (Gargoyle Games)</i>	8
3.	<i>ENTERFACE (Enterprise melléklet)</i>	15
4.	<i>Ismeretlen nyelvek (Micro-PROLOG: A perifériák kezelése)</i>	19
	- <i>KILLED UNTIL DEAD (Level III.)</i>	21
5.	<i>Ismeretlen nyelvek (HISOFT 'C' Compiler)</i>	22
6.	<i>Hardware ötletek (Numerikus billentyűzet)</i>	25
	- <i>KILLED UNTIL DEAD (Level IV.)</i>	26
7.	<i>BASIC (A billentyűzet figyelése, grafikai ötletek)</i>	27
8.	<i>Gépi kód tanfolyam</i>	28

SpV. 18.rész EP melléklet Mercenary



- Az úrsiklót nem szükséges megvenni, el is lophatjuk. Ekkor nagy valószínűséggel lelőnek, de sebaj, a CAPS SHIFT + Q-val új úrhajót kérhetünk.
 - Az úrállomáson lévő halálfejes ajtó nem halálos, csak lezuhanunk, és hosszú sétát kell tennünk egy hangárig, melyben van úrsikló, vagy segít a CAPS SHIFT + Q.
 - A célkereszt (sights) nagymértékben megkönnyíti az ajtókon való bejutást.
- A 03-15 szektorba könnyű eljutni, ha a 09-06 szektorban az orvosi felszerelésektől kimegyünk a folyosóra, és bemegyünk a szemközti ajtón. A második ajtó balra, és ha szerencsénk van, akkor a 03-15-ben kötünk ki, ha nem, akkor próbálkozunk tovább.

SpV. 14.rész 5. oldal Garfield

A sintértelep kulcsának megszerzésére van egy egyszerűbb módszer is: Menjünk el Nermál barátunkhoz, aki a csatornában van az egérrel. (Ha a csatornába levisszük a lámpát, az világít nekünk.) Nermált kb. úgy jó 5-6-szor rúgjuk fenékre, ekkor leejti a felhúzható egeret. Ezt kapjuk fel, és vigyük ki a csatornarendszerből. Ha tehát Nermál keze üres, rohangál a csatornarendszerben. Álljunk készen a láda melletti szobában, majd amikor megjelenik Nermál, induljunk el a ládához. Ahogy a ládához értünk, rúgjunk bele a ládába, ő felveszi a csontot, mi pedig felvehetjük a kulcsot. A gaz patkány, aki a ládát őrzi, nem tud beleszólni az akcióba.

SpV. 18.rész, EP.melléklet SPECTRUM programok átírása

A programlistába két helyen is hiba csúszott:

L1	LD	(HL),A	LD FLAG	RRA	és nem RRC
	INC	HL		XOR	L
	LD	A,(3FFFh)		RET	NZ
	CALL	-WRA			és nem WR

A hibák kijavítása után a program már jól fut.

Számszoba
 Budapest XIII., Sallai u. 28.
 személyi számítógépek
 és tartozékaik javítása
SINCLAIR SPECTRUM
 javítás 48 óra alatt
 6 hónap garancia
 NYITVA: hétfőtől - péntekig
 9.30-18 óráig.
 szombaton: zárva

PARAGE -
 EGA V04 PC/AT 286

DD DOLBY
 nagy választékban!

1114 Budapest
 Bercsényi ut. 7.
 Telefon: 612-046

NYITVA: hétfőtől - péntekig 9.30-18 óráig

PROGRAMAJÁNLAT
IBM PC
FELHASZNÁLÓKNAK

PC11
Megnevezés: VISIBLE PASCAL
File-ok száma: 35
Memóriafoglaltság (termézet): 182K
Besorolás: Nivós

Ez egy programcsomag, amely kiválóan alkalmas oktatási célra. A VISIBLE PASCAL a PASCAL egy jellegű nyelvre, amely nagy segítség lehet azoknak, akik csak most szeretnék megismerkedni ezzel a nyelvi formával. A program az oktatás során megmutatja, hogyan kell az ismeretlen nyelvi elemeket használni. A program a PASCAL nyelvet a programozás kezdő lépéseitől fogva tanítja meg. A program a felhasználó számára a programozás alapjait tanítja meg. A program a felhasználó számára a programozás alapjait tanítja meg.

PC12
Megnevezés: PC Graph
File-ok száma: 14
Memóriafoglaltság (termézet): 204K
Besorolás: Adaptív/Állászó

Ez a képfeldolgozó alkalmazás a számítógépes képfeldolgozás alapjait tanítja meg. A program a felhasználó számára a képfeldolgozás alapjait tanítja meg. A program a felhasználó számára a képfeldolgozás alapjait tanítja meg.

PC13
Megnevezés: COMPSER
File-ok száma: 30
Memóriafoglaltság (termézet): 143K
Besorolás: Zenehallgató

Ez a program a zenehallgatás alapjait tanítja meg. A program a felhasználó számára a zenehallgatás alapjait tanítja meg. A program a felhasználó számára a zenehallgatás alapjait tanítja meg.

PC14/PC15
Megnevezés: Generic Adventure Game System
File-ok száma: 1716
Memóriafoglaltság (termézet): 324,111K
Besorolás: Járvázaraktató

Ez a program a járvázaraktató alapjait tanítja meg. A program a felhasználó számára a járvázaraktató alapjait tanítja meg. A program a felhasználó számára a járvázaraktató alapjait tanítja meg.

PC16
Megnevezés: CALICO-PRINTER Utility
File-ok száma: 47
Memóriafoglaltság (termézet): 315K
Besorolás: Utility

Ez a program a CALICO-PRINTER alapjait tanítja meg. A program a felhasználó számára a CALICO-PRINTER alapjait tanítja meg. A program a felhasználó számára a CALICO-PRINTER alapjait tanítja meg.

PC17
Megnevezés: DRIGAM!
File-ok száma: 32
Memóriafoglaltság (termézet): 343K
Besorolás: Öltető

Ez a program az öltető alapjait tanítja meg. A program a felhasználó számára az öltető alapjait tanítja meg. A program a felhasználó számára az öltető alapjait tanítja meg.

PC18
Megnevezés: MACK'S DOMINO PARLOR
File-ok száma: 9
Memóriafoglaltság (termézet): 153K
Besorolás: Játék

Ez a program a MACK'S DOMINO PARLOR alapjait tanítja meg. A program a felhasználó számára a MACK'S DOMINO PARLOR alapjait tanítja meg. A program a felhasználó számára a MACK'S DOMINO PARLOR alapjait tanítja meg.

PC19
Megnevezés: MUSIC ARCHIVE
File-ok száma: 5
Memóriafoglaltság (termézet): 119K
Besorolás: Adatbázis/Állászó

Ez a program a MUSIC ARCHIVE alapjait tanítja meg. A program a felhasználó számára a MUSIC ARCHIVE alapjait tanítja meg. A program a felhasználó számára a MUSIC ARCHIVE alapjait tanítja meg.

PC20
Megnevezés: HARDDISK TESTER
File-ok száma: 11
Memóriafoglaltság (termézet): 136K
Besorolás: Utility

Ez a program a HARDDISK TESTER alapjait tanítja meg. A program a felhasználó számára a HARDDISK TESTER alapjait tanítja meg. A program a felhasználó számára a HARDDISK TESTER alapjait tanítja meg.

SPECTRUM VILÁG
 Budapest
 Pf.: 363
 1519

Az új, látnivaló IBM PC XT/AT programjainkban is megrendelhető a következő címen: **VILÁG**

Egy lemez ára: 500,- Ft (AFÁ-val és postaköltséggel együtt)

Több lemez egyidejű rendelése esetén kedvezményt adunk! 5 lemez ára: 2250,- míg 10 lemez ára már csak 4000,- Ft!

A közlemények célja, hogy szállításukkor a megrendeléseit egyelőre **feljuttatjuk!**

A MŰSZAKI KÖNYVKIADÓ könyvajánlata
A felsorolt könyvek megrendelhetők, ill. megvásárolhatók:
Műszaki Könyvkiadó Kandó Kálmán könyvesboltja,
Budapest V., Bajcsy-Zs. út 20. 1051

Könözy - Gál

Csupa szuperjáték C-PLUS/4, C-16, C-116

Kb. 156 oldal, 135 Ft

A könyv a Csupa játék sorozat legújabb kötete, amelyben játékok és felhasználói programok, valamint "tippek és trükkök" találhatók. A játékok látványos grafikával és kellemes zenével készülnek, jó szórakozást ígérnek. A felhasználói programok nagy segítséget nyújtanak a programíráshoz, a grafika előállításához, ill. programok másolásához. A programok magas színvonalúak, remélhetően sok örömet szereznek a gyakorlott és kezdő játékosok, valamint egyéb felhasználók számára.

Mazzag Mihály

LOTUS 1-2-3

(lapozgató sorozat)

Kb. 240 oldal, 220 Ft

A táblázatkezelő programok elterjesztésének és tényleges használatának gyorsítására mindent el kell követni, mert a felhasználó számára a számítástechnika hasznát ezek a programok teszik legközvetlenebbül hozzáférhetővé. A LOTUS 1-2-3 az egyik legismertebb táblázatkezelő program, de csak elvétve találkozhatunk magyar nyelvű leírásával. A lapozgató sorozat, amely az IBM PC-hez kapcsolódó programok tömör, pontos leírása példákkal és megjegyzésekkel kiegészítve, következő kötetével ezt a hiányt akarja pótolni.

A könyv mintegy 150 parancssorozatot, 80 függvényt és 50 makrót ismertet. Mindegyiknél megadja a formátumot, a verziószámot és alkalmazási lehetőségeit közöl. A lapozgató sorozat nem pótolja a dokumentációt, de olyan segédeszköz, amely minden felhasználónak gyorsan megtalálható, pontos információt ad.

Magyar István
OPEN ACCESS
(lapozgató sorozat)

Kb. 200 oldal, 180 Ft

Az OPEN ACCESS integrált szoftver a kevés számítástechnikai gyakorlattal rendelkező felhasználói kör számára olyan eszköz, amely széles körű szolgáltatásaival (adatbáziskezelés, számolótábla, szövegfeldolgozás, kalendárium, grafika, kommunikáció) képes az alkalmazói igények kielégítésére, kezelése pedig gyorsan és egyszerűen elsajátítható.

A lapozgató sorozat e kötete az alapvető tudnivalók összefoglalása után tömören, a formai előírások pontos betartásával ismerteti a programmodulok működését és a parancskészlet használatát. Külön fejezet foglalkozik a hibaüzenetekkel.

Kőhegyi János

Ismerd meg a BASIC nyelvjárásait!
(Commodore-16, Commodore PLUS/4,
Commodore-128, Videoton TV-Computer)

Kb. 144 oldal, 135 Ft

A BASIC nyelv a számítógépet használók körében közismert nyelv, amelynek géptípusonként sajátos változatai vannak. A könyv Donald Alcock: Ismerd meg a BASIC nyelvet c. közismert könyvére támaszkodva a Commodore-16, Commodore PLUS4, Commodore-128 és Videoton TV-Computer BASIC nyelvjárásait ismerteti.

Tartalom:

Előszó / C16 BASIC / Commodore PLUS4 BASIC / Commodore 128 BASIC / Videoton TV Computer BASIC

A 'SpV' 20. számában megjelent keresztrejtvény helyes megfejtései:

Válasz: EGYRI KEVESEBBEN, 76. TÖBLIAS KALANDJAI, Függy - ESPIONAGE ISLAND, 17. LE KELL FORDÍTANI

A 19. szám szerencsés nyertesei: HLA - Budapest XII. (S31), H.J. - Hatásvételek (C101), K.L. - Kerepestarton (S126), K.T. - Szigetzentmiklós (S95), V.B. - Budapest XXII. (S136), nyertesnévüket postálták!